Discrete Applied Mathematics 160 (2012) 1661–1679

Contents lists available at SciVerse ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

On the approximability of some degree-constrained subgraph problems☆

Omid Amini [a], David Peleg [b], Stéphane Pérennes [c], Ignasi Sau [d,*], Saket Saurabh [e]

[a] CNRS, Département de mathématiques et applications (DMA), École Normale Supérieure, Paris, France

[b] Weizmann Institute of Science, Rehovot, Israel

[c] Mascotte project INRIA/CNRS/UNS, Sophia-Antipolis, France

[d] CNRS, Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM), Montpellier, France

[e] The Institute of Mathematical Sciences, Chennai, India

ARTICLE INFO

Article history:
Received 29 November 2009
Received in revised form 12 October 2011
Accepted 21 March 2012
Available online 21 April 2012

Keywords:
Degree-constrained subgraph
Approximation algorithms
Hardness of approximation

ABSTRACT

In this article we provide hardness results and approximation algorithms for the following three natural degree-constrained subgraph problems, which take as input an undirected graph $G = (V, E)$. Let $d \geq 2$ be a fixed integer. The Maximum d − degree-bounded Connected Subgraph (MDBCS$_d$) problem takes as additional input a weight function $\omega : E \to \mathbb{R}^+$, and asks for a subset $E' \subseteq E$ such that the subgraph induced by E' is connected, has maximum degree at most d, and $\sum_{e \in E'} \omega(e)$ is maximized. The Minimum Subgraph of Minimum Degree $\geq d$ (MSMD$_d$) problem involves finding a smallest subgraph of G with minimum degree at least d. Finally, the Dual Degree-dense k-Subgraph (DDDkS) problem consists in finding a subgraph H of G such that $|V(H)| \leq k$ and the minimum degree in H is maximized.

1. Introduction

In this article, we consider three natural degree-constrained subgraph problems and study them in terms of approximation algorithms. A general instance of a degree-constrained subgraph problem [35,1,6] consists of an edge-weighted or vertex-weighted graph and the objective is to find an optimal weighted subgraph, subject to certain degree constraints on the vertices of the subgraph. The *degree* of a vertex v in a graph G, denoted $\deg_G(v)$, is the number of edges incident to v in G. We denote the maximum (respectively, minimum) vertex degree in the graph G by Δ_G (resp., δ_G).

Degree-constrained subgraph problems have attracted a lot of attention in the last decades and have resulted in a large body of literature [1,6,16,19–21,24,28,32,35,34]. The most well-studied ones are probably the Minimum-Degree Spanning Tree [19] and the Minimum-Degree Steiner Tree [20] problems. Beyond the esthetic and theoretical appeal of degree-constrained subgraph problems, the reasons for such intensive study are rooted in their wide applicability in the areas of interconnection networks and routing algorithms, among others. For instance, given an interconnection network modeled by an undirected graph, one may be interested in finding a small subset of nodes having a high degree of connectivity with the other nodes. This translates into finding a small subgraph with a lower bound on the degree of its vertices, i.e., to the MSMD$_d$ problem, to be defined shortly. Note that if the input graph is bipartite, these problems are equivalent to classical transportation and assignment problems in operations research.

☆ An extended abstract containing some of the results of this paper appeared in the Proceedings of WAOA 2008, volume 5426 of LNCS, pages 29–42.

* Corresponding author. Tel.: +33 34626423588; fax: +33 33467418500.
E-mail addresses: omid.amini@ens.fr (O. Amini), david.peleg@weizmann.ac.il (D. Peleg), stephane.perennes@sophia.inria.fr (S. Pérennes), ignasi.sau@lirmm.fr (I. Sau), saket@imsc.res.in (S. Saurabh).

0166-218X/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2012.03.025

The first problem studied in the paper is a classical NP-hard problem listed in [23] (cf. Problem [GT26] for the unweighted version). Let $d \geq 2$ be a fixed integer.

> MAXIMUM d-DEGREE-BOUNDED CONNECTED SUBGRAPH (MDBCS$_d$)
> **Input**: A graph $G = (V, E)$ and a weight function $\omega : E \to \mathbb{R}^+$.
> **Output**: A subset $E' \subseteq E$ such that the subgraph $G' = (V, E')$ is connected (except, possibly, for isolated vertices), has maximum degree at most d, and $\sum_{e \in E'} \omega(e)$ is maximized.

For $d = 2$, the unweighted MDBCS$_d$ problem corresponds to the LONGEST PATH problem. Indeed, given the input graph G (which can be assumed to be connected), let P and G' be optimal solutions of LONGEST PATH and MDBCS$_2$ in G, respectively. Then observe that $|E(G')| = |E(P)|$ unless G is Hamiltonian, in which case $|E(G')| = |E(P)| + 1$. One could also ask the question: what happens when G' is not required to be connected in the definition of MDBCS$_d$? It turns out that without the connectivity constraint, both the edge version and the vertex version (where the goal is to maximize the total weight of the vertices of a subgraph satisfying the degree constraints) of the MDBCS$_d$ problem are known to be solvable in polynomial time using matching techniques [11,23,27]. In fact, without connectivity constraints, even a more general version where the input contains an interval of allowed degrees for each node is known to be solvable in polynomial time.

For a finite, simple, and undirected graph $G = (V, E)$ and $d \in \mathbb{N}$, the *d-girth* of G is the minimum number of vertices of an induced subgraph of G of minimum degree at least d. The notion of d-girth was proposed and studied by Erdős et al. [15,16] and Bollobás and Brightwell [10]. It generalizes the usual girth, the length of a shortest cycle, which coincides with the 2-girth. (This is indeed true because every subgraph of minimum degree at least two contains a cycle.) Combinatorial bounds on the d-girth can also be found in [25,7]. We are unaware of complexity results of the corresponding optimization problem. In an attempt to fill this void in the literature, we define the following problem for $d \geq 2$ being a fixed integer. (For a graph $G = (V, E)$ and $S \subseteq V$, we denote by $G[S]$ the induced subgraph of G with vertex set S.)

> MINIMUM SUBGRAPH OF MINIMUM DEGREE $\geq d$ (MSMD$_d$)
> **Input**: An undirected graph $G = (V, E)$.
> **Output**: A subset $S \subseteq V$ such that for $H = G[S]$, $\delta_H \geq d$ and $|S|$ is minimized.

Note that the MSMD$_d$ problem is in P for $d = 2$, as it is exactly the GIRTH problem. We shall see that the situation is quite different for $d \geq 3$. Note also that MSMD$_d$ can be viewed as a dual (unweighted) node-minimization version of MDBCS$_d$. Another motivation for studying MSMD$_d$ is its close relation to the well studied DENSE k-SUBGRAPH (DkS) [18,26,8] and TRAFFIC GROOMING [4] problems. See [4,5] for further details. Recently, Amini et al. [5] studied the MSMD$_d$ problem in the realm of parameterized complexity. The authors provided W[1]-hardness results for general graphs and explicit *fixed-parameter tractable* (FPT) algorithms for the class of graphs excluding a fixed graph as a minor and for graphs of bounded local tree-width. We note that if in the definition of MSMD$_d$ we replace "minimized" with "maximized", then the objective subset S is known as a *d-core*, and can be easily found by recursively removing vertices of degree less than d.

The last problem studied in this paper is a natural variation of the MSMD$_d$ problem, in which instead of minimizing the size of a subgraph for a given minimum degree, we aim at maximizing the minimum degree of a subgraph of a given size.

> DUAL DEGREE-DENSE k-SUBGRAPH (DDDkS)
> **Input**: An undirected graph $G = (V, E)$ and a positive integer k.
> **Output**: An induced subgraph H of size $|V(H)| \leq k$, such that δ_H is maximized.

Note that the NP-hardness of DDDkS easily follows from MAXIMUM CLIQUE. Indeed, the optimal to the DDDkS problem is $k - 1$ if and only if G has a clique of size k.

The above discussion illustrates that the study of these problems is very natural and that the results obtained for them can reverberate in several other important optimization problems, coming from both theoretical and practical domains.

Our results. In this paper we obtain both approximation algorithms and results on hardness of approximation. All of our hardness results are based on the hypothesis that P $\neq$ NP. More precisely, our results are the following:

- We prove that the MDBCS$_d$ problem is not in APX for any $d \geq 2$, and that if there is a polynomial-time algorithm for MDBCS$_d$, $d \geq 2$, with approximation ratio $2^{\mathcal{O}(\sqrt{\log n})}$, then NP $\subseteq$ DTIME$(2^{\mathcal{O}(\log^5 n)})$. These hardness results hold also for unweighted graphs. On the other hand, we give an approximation algorithm for general unweighted graphs with ratio $\min\{m/\log n,\ nd/(2\log n)\}$, and an approximation algorithm for general weighted graphs with ratio $\min\{n/2,\ m/d\}$. The first algorithm uses an algorithm introduced in [3], which is based on the *color-coding* method. We also present a constant-factor approximation when the input graph has a low-degree spanning tree, in terms of the integer d.

- We prove that the MSMD$_d$ problem is not in Apx for any $d \geq 3$. The proof is obtained by the following two steps. First, by a reduction from Vertex Cover in regular graphs, we prove that MSMD$_d$ does not admit a PTAS. In particular, this implies that MSMD$_d$ is NP-hard for any $d \geq 3$. Then, we use the *error amplification* technique to prove that MSMD$_d$ is not in Apx for any $d \geq 3$. On the positive side, we give an $(n/\log n)$-approximation algorithm for the class of graphs excluding a fixed graph H as a minor, using a known structural result on graph minors and dynamic programming over graphs of bounded tree-width. In particular, this gives an $(n/\log n)$-approximation algorithm for planar graphs and graphs of bounded genus.
- We observe that an α-approximation algorithm for the Dense k-Subgraph problem can be turned into a 2α-approximation algorithm for the DDDkS problem. This fact implies, according to a recent result of Chlamtac and Feige [8], the existence of an algorithm that for every $\varepsilon > 0$ approximates the DDDkS problem within a ratio of $n^{1/4+\varepsilon}$ in time $n^{\mathcal{O}(1/\varepsilon)}$. We also provide a simple randomized $\mathcal{O}(\sqrt{n \log n})$-approximation algorithm, which does not use any "black-box" as a subroutine.

Finally, we would like to point out the large gap between the hardness results and the approximation ratios of the algorithms presented in this article. Although it may be possible to obtain better approximation ratios for the above three problems, we suspect most of the approximation ratios to be not far from the optimal.

Organization of the paper. Section 2 provides some basic definitions required in the paper. In Section 3 we establish inapproximability results for MDBCS$_d$ for any $d \geq 2$, and in Section 4 we present two approximation algorithms for unweighted and weighted general graphs, respectively. The constant-factor approximation for MDBCS$_d$ when the input graph has a low-degree spanning tree is provided in Section 4.2. In Section 5 we prove that MSMD$_d$ is not in Apx for any $d \geq 3$, and in Section 6 we give an $(n/\log n)$-approximation algorithm for the class of graphs excluding a fixed graph H as a minor. In Section 7 we focus on approximation algorithms for the DDDkS problem. Finally, we conclude with some remarks and open problems in Section 8.

2. Basic definitions

For the sake of completeness, we provide in this section some basic definitions to be freely used throughout the paper. For additional background material, the reader is referred, for example, to [36]. Unless explicitly stated otherwise, log denotes logarithm to the base two.

Given an NP-hard minimization (resp. maximization) problem Π and a polynomial-time algorithm $\mathcal{A}$, let $\mathrm{OPT}_\Pi(I)$ be the optimal value of the problem Π for the instance I, and let $\mathrm{ALG}(I)$ be the value given by algorithm $\mathcal{A}$ for the instance I. We say that $\mathcal{A}$ is an *α-approximation algorithm* (or an approximation algorithm with ratio α) for Π if for any instance I of Π, $\mathrm{ALG}(I)/\mathrm{OPT}_\Pi(I) \leq \alpha$ (resp. $\mathrm{OPT}_\Pi(I)/\mathrm{ALG}(I) \leq \alpha$). Note that $\alpha \geq 1$.

The class Apx consists of all NP-hard optimization problems that can be approximated within a constant factor. The subclass PTAS (standing for Polynomial Time Approximation Scheme) contains the problems that can be approximated in polynomial time within a ratio $1 + \varepsilon$ for *any* constant $\varepsilon > 0$. Assuming P $\neq$ NP, there is a strict inclusion of PTAS in Apx (for instance, Vertex Cover is in Apx $\setminus$ PTAS), hence an Apx-hardness result for a problem implies the non-existence of a PTAS. Let us recall the definitions of the Minimum Vertex Cover problem (from which we obtain the hardness reduction of Section 5) and the Dense k-Subgraph problem.

Vertex Cover (VC)
Input: An undirected graph $G = (V, E)$.
Output: A subset $S \subseteq V$ of the minimum size such that for each edge $\{u, v\} \in E$, at least one of u and v belongs to S.

Dense k-Subgraph (DkS)
Input: An undirected graph $G = (V, E)$ and a positive integer k.
Output: A subset $S \subseteq V$, with $|S| = k$, such that $|E(G[S])|$ is maximized.

Definition 2.1 (*Tree-Decomposition, Tree-Width*)**.** A *tree-decomposition* of a graph $G = (V, E)$ is a pair $(T, \mathcal{X})$, where $T = (I, F)$ is a tree, and $\mathcal{X} = \{X_i \mid i \in I\}$ is a family of subsets of $V(G)$, called *bags* and indexed by the nodes of T, such that

(1) each vertex $v \in V$ appears in at least one bag, i.e., $\bigcup_{i \in I} X_i = V$;
(2) for each $v \in V$ the set of nodes indexed by $\{i \mid i \in I,\ v \in X_i\}$ forms a subtree of T;
(3) For each edge $e = \{x, y\} \in E$, there is an $i \in I$ such that $x, y \in X_i$.

The *width* of a tree-decomposition is defined as $\max_{i \in I}\{|X_i| - 1\}$. The *tree-width* of G, denoted by $tw(G)$, is the minimum width of a tree-decomposition of G.

3. Hardness of approximating MDBCS$_d$

As mentioned in Section 1, MDBCS$_2$ corresponds basically to the Longest Path problem, which is known to not admit any constant-factor approximation [24], unless P = NP. In this section we extend this result and prove that, under the assumption that P ≠ NP, MDBCS$_d$ is not in Apx for any $d \geq 2$, proving first that MDBCS$_d$ is not in PTAS for any $d \geq 2$. Finally, we also prove in Theorem 3.4 that if there is a polynomial time algorithm for MDBCS$_d$, $d \geq 2$, with an approximation ratio of $2^{\mathcal{O}(\sqrt{\log n})}$, then NP $\subseteq$ DTIME$(2^{\mathcal{O}(\log^5 n)})$. In the remainder of this section we focus on the cases $d \geq 3$, as the case $d = 2$ follows from [24]. We note that our hardness results hold even when all edges have unitary weight, as it is the case for the Longest Path problem.

Theorem 3.1. MDBCS$_d$ *does not admit a* PTAS *for any* $d \geq 3$, *unless* P = NP.

Proof. We give our reduction from the Traveling Salesman problem with two distinct weights on the edges, namely TSP(1, 2), which does not admit a PTAS unless P = NP [31]. An instance of TSP(1, 2) consists of a complete graph $G = (V, E)$ on n vertices and a weight function $f : E \to \{1, 2\}$ on its edges, and the objective is to find a traveling salesman tour of minimum weight in G.

We show that if there is a PTAS for MDBCS$_d$, for a fixed $d \geq 3$, then one can construct a PTAS for TSP(1, 2). Towards this, we transform the graph G into a new augmented graph G' with a modified weight function g on its edges. For every vertex $v \in V$ we add to G' $d-2$ new vertices $\{v_1, \ldots, v_{d-2}\}$ and an edge from v to every vertex v_i, $1 \leq i \leq d-2$. Thus $G' = (V \cup V', E \cup E')$, where $V' = \bigcup_{v \in V}\{v_1, \ldots, v_{d-2}\}$ is the set of new vertices and $E' = \{\{v_i, v\} \mid 1 \leq i \leq d-2, v \in V\}$ is the set of new edges. We define the weight function g on the edges of G' as:

$$g(e) = \begin{cases} 3 - f(e), & e \in E, \text{ (weights of original edges get flipped)} \\ 3, & e \in E'. \end{cases}$$

See Fig. 1(a) for an example of the constructed graph G' and the weight function g with $n = d = 4$. Next we prove a claim characterizing the structure of the *maximal* solutions of MDBCS$_d$ in G'. Essentially, it shows that any given solution G_1 of MDBCS$_d$ in G' with value W can be transformed into another solution G_2 of MDBCS$_d$ in G' with value at least W, such that G_2 contains all the newly added edges and induces a Hamiltonian cycle in G.

Claim 3.1. *Any given solution* $G_1 = (V \cup V', E_1)$ *of* MDBCS$_d$ *in* G' *can be transformed in polynomial time into a solution* $G_2 = (V \cup V', E_2)$ *of* MDBCS$_d$ *in* G' *such that* (i) $G_3 = (V, E \cap E_2)$ *is a Hamiltonian cycle in* G, *and* (ii) $\sum_{e \in E_2} g(e) \geq \sum_{e' \in E_1} g(e')$.

Proof. We prove the claim by describing a series of transformations, applied in order of appearance, successively improving the solution, and eventually yielding the desired G_2. For a given edge set F, let $X(F)$ be the set of vertices containing the end-vertices of the edges in F.

(a) Suppose $E_1 \cap E' = \emptyset$. Then $H = (X(E_1), E_1)$ is connected and every vertex $v \in X(E_1)$ has degree at most d in H. If H has some vertex v of degree strictly less than d, we can add to the solution the edge $\{v_1, v\}$. Otherwise, all vertices in H have degree exactly d. In particular, H contains a cycle, so removing any edge from this cycle will not break the connectivity of the solution. So we can remove any edge $\{u, v\}$ from this cycle and add the edges $\{u_1, u\}$ and $\{v_1, v\}$, obtaining a solution of larger weight. Therefore, we assume henceforth that $E_1 \cap E' \neq \emptyset$.

(b) Suppose $V \setminus X(E_1) \neq \emptyset$, that is, there is a vertex $v \in V$ which is not contained in $X(E_1)$. In this case, by case (a) there exists a vertex $u \in X(E_1)$ such that one of the edges $\{u_i, u\}$, $1 \leq i \leq d-2$, is in E_1. We then set $E_1 \leftarrow E_1 - \{\{u_i, u\}\} \cup \{\{u, v\}, \{v, v_i\} \mid 1 \leq i \leq d-2\}$. Clearly, connectivity is maintained (as removing edges from E' does not break connectivity) and the weight of solution increases by at least 1. This procedure is repeated until the current solution contains all the vertices of G.

(c) Suppose $H' = (V, E \cap E_1)$ is neither a spanning tree nor a Hamiltonian cycle. Notice that H' is connected, as removing degree 1 vertices of V' does not break connectivity. This implies that there is a cycle C in H' and a vertex v on it such that $\deg_{H'}(v) \geq 3$ (otherwise, H' would be disconnected). This implies that there exists an edge $e = \{v, v_i\}$ such that $e \notin E_1$. Let $\{u, v\}$ be an edge on C. We then set $E_1 \leftarrow E_1 - \{\{u, v\}\} \cup \{\{v, v_i\}\}$. Again, connectivity is clearly maintained (as removing an edge from a cycle does not break connectivity) and the weight of the solution increases by at least 1. This procedure is repeated until H' is either a spanning tree or a Hamiltonian cycle.

(d) Suppose $H' = (V, E \cap E_1)$ is a spanning tree. We take any two leaves u and v of H' and add the edge $\{u, v\}$, obtaining a solution of greater weight. If the obtained graph is a Hamiltonian cycle, we are done, otherwise we go back to case (c).

The above transformation rules can be applied in polynomial time to obtain a graph G_3 that is a solution of MDBCS$_d$ in G' and satisfies the conditions described in the statement of the claim. □

Suppose that there exists a PTAS for MDBCS$_d$ realized by an approximation scheme $\mathcal{A}_\delta$. This family of algorithms takes as input a graph G' and a parameter $\delta > 0$, and returns a solution of MDBCS$_d$ of weight at least $(1 - \delta)\text{OPT}_{G'}$, where $\text{OPT}_{G'}$ is the value of an optimal solution of MDBCS$_d$ in G'.

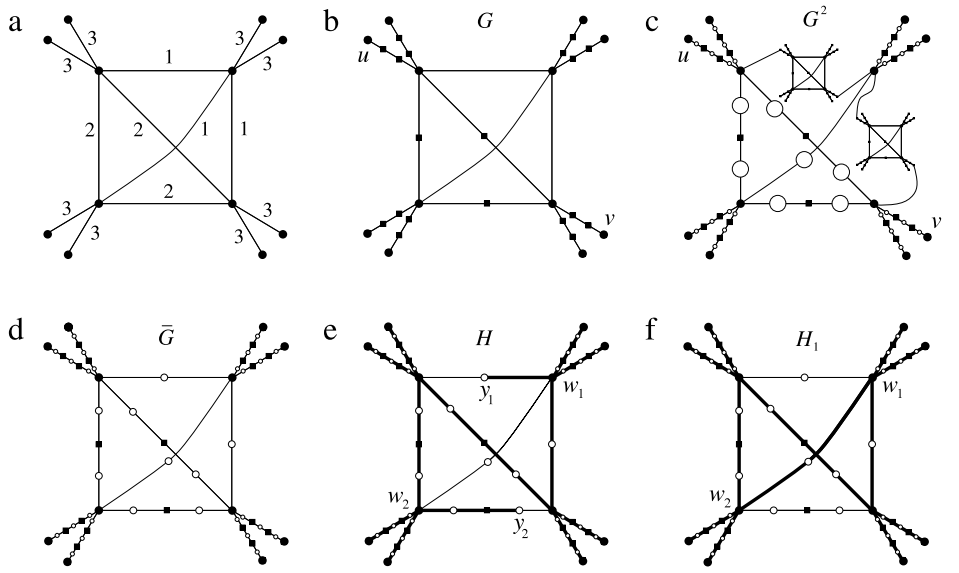

Fig. 1. Example of the transformations for $d = 4$. (a) Graph obtained from an instance of TSP(1, 2) with $n = d = 4$. The integers correspond to the weight function g. (b) Subdivided graph $G \in \mathcal{G}$ obtained from the instance of TSP(1, 2). The ribbon edge $\{u, v\}$ has been omitted in the figure. (c) Graph G^2 obtained from G, where for better visualization each circle corresponds to a copy of G. (d) Graph $\bar{G}$ obtained from G^2. (e) The thick edges define a solution H in $\bar{G}$. (f) The thick edges define a solution H_1 in $\bar{G}$ obtained from H.

Using this scheme, we now proceed to construct a PTAS for TSP(1, 2). Given a graph G, an instance of TSP(1, 2), and a real number $\varepsilon > 0$, do the following:

(1) Apply the transformation described before Claim 3.1 to G and obtain the graph G'.
(2) Fix $\delta = h(\varepsilon, d)$ (to be specified later) and run $\mathcal{A}_\delta$ on G'. Let G'' be the resulting solution.
(3) Apply the polynomial-time transformation described in Claim 3.1 on G'', the solution obtained by $\mathcal{A}_\delta$ on G'. Let the new solution be $G^* = (V \cup V_1, E^*)$.
(4) Return $E^* \cap E$ as the computed solution of TSP(1, 2).

Now we prove that the solution returned by our algorithm satisfies $\sum_{e \in E^* \cap E} f(e) \leq (1 + \varepsilon)O_T$, where O_T is the weight of an optimal tour in G. Let such an optimal tour contain a edges of weight 1 and b edges of weight 2. Then $O_T = a + 2b$ and $a + b = n$. Equivalently $a = 2n - O_T$ and $b = O_T - n$. Let O_D be the value of an optimal solution of MDBCS$_d$ in G'. Then by Claim 3.1 and the flipping nature of the function g, we have that

$$O_D = (d - 2)3n + 2a + b. \tag{1}$$

Let $3(d-2)n + O_D^*$ be the value of the solution returned by $\mathcal{A}_\delta$, where O_D^* is the sum of the edge weights of the Hamiltonian cycle in G^*, that is, $O_D^* = \sum_{e \in E^* \cap E} g(e)$. Since $\mathcal{A}_\delta$ is a PTAS,

$$3(d - 2)n + O_D^* \geq (1 - \delta)O_D. \tag{2}$$

Combining Eq. (1) and Inequality (2) gives

$$O_D^* \geq (1 - \delta)O_D - 3(d - 2)n = 3n - O_T + \delta O_T - n(3d - 3)\delta. \tag{3}$$

On the other hand, the value of the solution returned by our algorithm for TSP(1, 2) is $O_T^* = 3n - O_D^*$ (since if $O_D^* = 2x + y$, x being the number of edges of weight 2 and y being the number of edges of weight 1, with $x + y = n$, then the value of the solution of TSP(1, 2) is $x + 2y$). Substituting $O_D^* = 3n - O_T^*$ in Inequality (3) and using the fact that $O_T \geq n$ yields

$$O_T^* \leq O_T - \delta O_T + n(3d - 3)\delta \leq O_T - \delta O_T + O_T(3d - 3)\delta = O_T + (3d - 4)\delta O_T. \tag{4}$$

To show that $O_T^* \leq (1 + \varepsilon)O_T$, by Eq. (4) it is enough to set $\delta = h(\varepsilon, d) = \frac{\varepsilon}{3d-4}$, yielding a PTAS for TSP(1, 2). Since TSP(1, 2) does not admit a PTAS [31], the last assertion also rules out the existence of a PTAS for MDBCS$_d$ for any $d \geq 3$, unless P = NP. □

We need some extra notation for the remainder of this section. By *subdividing* an edge $e = \{u, v\}$ of a graph we denote the operation of deleting the edge $\{u, v\}$, and adding a new vertex z together with the edges $\{u, z\}$ and $\{z, v\}$. There is a natural (maybe not unique) bipartition of the vertices of a subdivided graph G into *original* and *inner* vertices, where the original vertices are the vertices of the graph from which G has been obtained.

Corollary 3.1. *For any fixed integer $d \geq 3$,* MDBCS$_d$ *does not admit a* PTAS *even if all edges have unitary weight, unless* P = NP.

Proof. For any fixed integer $d \geq 3$, we slightly modify the reduction from TSP(1, 2) presented in the proof of Theorem 3.1. We transform the constructed graph G', with weight function $g : E(G') \to \{1, 2, 3\}$, into a graph G'' with unitary weights obtained from G' by subdividing each edge $e \in E(G')$ exactly $g(e) - 1$ times; see Fig. 1(a) and (b) for an example with $n = d = 4$. It can then be routinely checked that the arguments of the proof of Theorem 3.1 carry over to G'', just by replacing the role of each edge $e \in E(G')$ with weight $g(e)$ by the corresponding induced path in G'' with $g(e)$ edges. (It is safely understood that the notions of spanning tree and Hamiltonian cycle in the graph G translate into a tree and a cycle in G'', respectively, visiting all the original vertices of G.) □

From now on we will only deal with graphs with unitary edge weights. Given a subdivided graph G as defined in the proof of Corollary 3.1, each induced path of G with 3 edges obtained from an edge of weight 3 is called a *pendant path*, and the vertex of degree more than 2 to which this path is attached is called the *root* of the pendant path. Let u and v be two arbitrary (but fixed) end-vertices of two pendant paths with distinct roots (see Fig. 1(b) for an example with $d = 4$). We further modify G by adding the edge $\{u, v\}$, which we call a *ribbon edge*. For simplicity, in the notation we omit the choice of the vertices u, v.

Let $\mathcal{G}$ be the class of (unweighted) subdivided graphs defined in the proof of Corollary 3.1 by adding the corresponding ribbon edges, that is, the graphs obtained from instances of TSP(1, 2) by applying the transformations described above. The following fact is an immediate consequence of Corollary 3.1, by observing that ribbon edges do not alter the structure of the solutions of MDBCS$_d$ for $d \geq 3$, as we may assume that any solution contains all the roots and pendant paths, and therefore also the ribbon edge.

Corollary 3.2. *For any fixed integer $d \geq 3$,* MDBCS$_d$ *does not admit a* PTAS *in the class of graphs $\mathcal{G}$, unless* P = NP.

To show the non-existence of a constant-factor approximation for a fixed integer $d \geq 3$, we also need to introduce an *edge squaring* operation, starting from graphs in the class $\mathcal{G}$. We define G^2 as the graph obtained from $G \in \mathcal{G}$ by replacing every edge $e = \{x, y\} \in E$, except for the ribbon edge $\{u, v\}$, with a copy G_e of G, and adding the two edges $\{x, u\}$ and $\{y, v\}$; see Fig. 1(c) for an example with $d = 4$, where the white circles represent other copies of G. For better visibility, ribbon edges have been omitted in the figure. The vertices x and y are referred to as the *contact vertices* of G_e, and the edges $\{x, u\}$ and $\{y, v\}$ as the *contact edges* of G_e. The two contact edges corresponding to the same copy of G are called *twins*. When applying this operation iteratively to obtain graphs $G^4, G^8, \ldots, G^{2^p}$, we do *not* replace the contact edges and the ribbon edges with a copy of the current graph. That is, a contact or ribbon edge that has appeared at some stage of the edge squaring operation remains unchanged when further squaring the graph. A *p-contact edge* is a contact edge that has appeared when constructing G^{2^p} from $G^{2^{p-1}}$. Also, the contact edges corresponding to a copy of the current graph always contain one of the two original vertices u and v of G. For instance, in order to obtain G^4 from the graph G^2 of Fig. 1(d), each copy of G^2 is attached to the rest of the graph through either u or v. We denote by $\mathcal{G}^*$ the class of graphs that can be obtained from some graph in $\mathcal{G}$ by repeatedly applying the edge squaring operation.

Observe that this edge squaring differs from the one introduced in [24] to prove the hardness of Longest Path, in which for every edge $e = \{x, y\} \in E$, the vertices x and y are joined to every vertex in G_e. We need this new definition for technical reasons, as the structure of the solutions of MDBCS$_d$ for $d = 2$ and for $d \geq 3$ differs considerably.

The squared graphs G^{2^p} are important because of the following facts.

Lemma 3.1. *Let $d \geq 3$ and $p \geq 1$ be two integers. Let $G \in \mathcal{G}$ be a graph in which an optimal solution of* MDBCS$_d$ *has x edges. Then an optimal solution of* MDBCS$_d$ *in G^{2^p} has at least $x^{2^p} - y_p$ edges, where $y_p = \mathcal{O}(x^{2^p-1})$.*

Proof. For simplicity, let x_i be the number of edges of an optimal solution of MDBCS$_d$ in G^{2^i}, so $x_0 = x$. We prove the following stronger claim by induction on i: for every integer $i \geq 1$, there exists a solution in G^{2^i} with at least $x^{2^i} - y_i$ edges, and having at most z_i contact or ribbon edges, with $y_i = \mathcal{O}(x^{2^i-1})$ and $z_i = \mathcal{O}(x^{2^i-1})$. For $i = 1$, let S_0 be a solution in G with $x_0 = x$ edges, and note that by the proof of Claim 3.1, we can assume that all pendant paths of G, as well as the ribbon edge $\{u, v\}$, belong to S. Let S_1 be the solution in G^2 containing a copy of S for each edge of S which is not a contact or a ribbon edge, plus the corresponding contact vertices and paths. As all pendant paths belong to S and $d \geq 3$, the graph S_1 is well-defined. As all edges of S get squared, except for the ribbon edge, it holds that $|E(S_1)| = 1 + (x - 1)(x + 2) = x^2 + x - 1 = x^2 - y_1$. Clearly, $y_1 = \mathcal{O}(x)$. The number of contact or ribbon edges of S_1 is $z_1 = 1 + (x - 1)(1 + 2) = 3x - 2 = \mathcal{O}(x)$.

Suppose now by induction that the claim is true for i, that is, there exists a solution S_i in G^{2^i} with $x^{2^i} - y_i$ edges having at most z_i contact or ribbon edges, with $y_i = \mathcal{O}(x^{2^i-1})$ and $z_i = \mathcal{O}(x^{2^i-1})$. Analogously to the case $i = 1$, when we square the edges belonging to S_i, except for contact and ribbon edges, replacing each of them with a copy of S_i, we obtain a solution S_{i+1} in $G^{2^{i+1}}$ satisfying

$$\begin{aligned} |E(S_{i+1})| &= z_i + (x^{2^i} - y_i - z_i)(x^{2^i} - y_i + 2) \\ &= x^{2^{i+1}} - x^{2^i}(y_i - 2) - (y_i + z_i)(x^{2^i} - y_i + 2) + z_i. \end{aligned} \tag{5}$$

As by the induction hypothesis it holds that $y_i = \mathcal{O}(x^{2^i-1})$ and $z_i = \mathcal{O}(x^{2^i-1})$, it follows from Eq. (5) that $x_{i+1} \geq x^{2^{i+1}} - y_{i+1}$, with $y_{i+1} = \mathcal{O}(x^{2^{i+1}-1})$.

On the other hand, the number of contact or ribbon edges of S_{i+1} is

$$z_{i+1} = z_i + (x^{2^i} - y_i - z_i)(z_i + 2). \tag{6}$$

Again, using that $y_i = \mathcal{O}(x^{2^i-1})$ and $z_i = \mathcal{O}(x^{2^i-1})$, it follows from Eq. (6) that $z_{i+1} = \mathcal{O}(x^{2^{i+1}-1})$, as we wanted to prove. □

The following lemma will play a fundamental role in the proof of Lemma 3.3.

Lemma 3.2. *Let $d \geq 3$ and $p \geq 1$ be two fixed integers, and let $G \in \mathcal{G}$. Given a solution S of* MDBCS$_d$ *in G^{2^p} with ℓ edges, we can transform it in polynomial time into another solution S' with $\ell - o(\ell)$ edges such that a contact edge belongs to S' if and only if its twin contact edge does.*

Proof. Let S be a solution of MDBCS$_d$ in G^{2^p} with ℓ edges. For $i = 1, \ldots, p$, we will sequentially transform S in polynomial time into a solution S_i of MDBCS$_d$ in G^{2^p} with the following property: $|E(S_i)| = \ell - o(\ell)$, and in S_i all j-contact edges with $j \leq i$ come in pairs, that is, a j-contact edge, with $j \leq i$, belongs to S_i if and only if its twin contact edge does. By letting $S' := S_p$ we obtain a solution with the claimed property. We now proceed to describe the transformation for $i = 1$.

For each pair of twin 1-contact edges e_1 and e_2 of G^{2^p}, the smallest connected component of $G^{2^p} \setminus \{e_1, e_2\}$ is called a *piece* of G^{2^p}. For instance, each circle of the graph depicted in Fig. 1(c) is a piece of G^2. Note that, in particular, by construction each piece is connected. Let $\bar{G}$ be the graph obtained from G^{2^p} by contracting each piece to a single vertex. Note that $\bar{G}$ is the graph obtained from G by subdividing each edge exactly once; see Fig. 1(d) for an example with $d = 4$. For simplicity, the vertices of $\bar{G}$ that correspond to pieces will be called *white vertices*. Also, an induced path in $\bar{G}$ between two original vertices not visiting any other original vertex is called a *direct path*.

Let X be the edges of $\bar{G}$ that belong to S, and let $H = \bar{G}[X]$, that is, the subgraph of $\bar{G}$ induced by the edges and vertices in X. Note that the edges in X are contact edges of G^{2^p}, and that by construction H is a solution of MDBCS$_d$ in $\bar{G}$; see Fig. 1(e) for an example with $d = 4$, where the subgraph H is defined by the thick edges. We now proceed to modify H into another solution H_1 of MDBCS$_d$ in $\bar{G}$ in which twin contact edges come in pairs. Finally, we will obtain from H_1 the desired solution S_1 of MDBCS$_d$ in G^{2^p}.

In the same spirit of the proof of Claim 3.1, we describe a series of transformations, applied to H in order of appearance, which eventually yield the desired solution H_1. During these transformations, we will make sure that the number of edges and white vertices of H does not decrease too much. Let E' be the set of edges of $\bar{G}$ that arose from subdividing an edge from a pendant path of G, let V_o be the set of original vertices of G contained in H, let $w(H)$ be the number of white vertices in H, and let $\hat{H}$ be the graph obtained from H be removing the vertices and edges belonging to pendant paths.

(a) Suppose $E(H) \cap E' = \emptyset$. If H is a tree (note that H is necessarily connected), let x be a leaf of H, and let y be the original vertex of $\bar{G}$ closest to x in H (it may be x itself). We remove from H the (possibly empty) path from y to x, and add to H a pendant path rooted at y. Clearly, connectivity and maximum degree are preserved, and both $|E(H)|$ and $w(H)$ strictly increase. Otherwise, if H contains a cycle C, let x and y be two consecutive original vertices in C (note that each cycle in H contains at least 3 original vertices). In this case, we remove from H the direct path from x to y (note that connectivity is preserved), and add two pendant paths rooted at x and y. After this transformation, $|E(H)|$ and $w(H)$ have increased by at least 8 and 4, respectively. We can assume henceforth that H contains at least an entire pendant path.

(b) Suppose $V_o \setminus V(H) \neq \emptyset$, that is, there is an original vertex x which is not contained in $V(H)$. In this case, by transformation (a) there exists a vertex $y \in V(H)$ such that one pendant path P_y rooted at y belongs to H. We remove P_y from H, and add the direct path from x to y (it may happen that part of this path was already in H; in that case we just make it longer until reaching x), and a pendant path rooted at x. Again, both $|E(H)|$ and $w(H)$ can only increase, and the connectivity and the maximum degree of H are preserved. This procedure is repeated until the current solution contains all the original vertices of G.

From now on, by a "spanning tree" (resp. "Hamiltonian cycle") we mean, with abuse of notation, a tree (resp. cycle) in $\bar{G}$ containing all original vertices of G.

(c) Suppose $\hat{H}$ is neither a spanning tree nor a Hamiltonian cycle. Notice that $\hat{H}$ is connected, as removing vertices of degree 1 preserves connectivity. This implies that there is a cycle C in $\hat{H}$ and a vertex x on it such that $\deg_{\hat{H}}(x) \geq 3$ (as otherwise, $\hat{H}$ would be disconnected). This implies that there is a pendant path P_x rooted at x which is not in H. Let y be an original vertex consecutive to x in C. We remove from H the direct path from x to v, and add the pendant path P_x. In this case, both $|E(H)|$ and $w(H)$ have strictly increased, and the connectivity and the maximum degree of H are preserved. This procedure is repeated until $\hat{H}$ is either a spanning tree or a Hamiltonian cycle.

(d) Suppose $\hat{H}$ is a spanning tree with a vertex x such that $\deg_{\hat{H}}(x) \geq 3$, and let $T_1, \ldots, T_\ell$ be the subtrees of $\hat{H}$ rooted at x. As $\deg_{\hat{H}}(x) \geq 3$, there is a pendant path P_x rooted at x which is not in $\hat{H}$. If only one of the trees $T_1, \ldots, T_\ell$ contains original vertices (other than x), then we can exchange each of them for a pendant path rooted at x, therefore increasing both $|E(H)|$ and $w(H)$. So we can assume that each subtree rooted at x contains at leat one original vertex. Let y_1 and y_2 be two leaves of $\hat{H}$ in the subtrees T_1 and T_2, respectively, and let w_1 and w_2 be the two original vertices which are

closest in $\hat{H}$ to y_1 and y_2, respectively (it may happen that $w_1 = y_1$ or $w_2 = y_2$). We remove from H the paths from y_1 to w_1 and from y_2 to w_2, add the direct path from w_1 to w_2, remove the edge of T_1 incident to x, and add the pendant path P_x rooted at x. In the new solution, the degree of x in $\hat{H}$ has decreased by one, H is still a spanning tree with $\Delta_H \leq d$, and one can check that $w(H)$ has not decreased and that $|E(H)|$ has increased by at least 1. This procedure is repeated until $\hat{H}$ satisfies $\Delta_{\hat{H}} \leq 2$, that is, until $\hat{H}$ is either a spanning path or a Hamiltonian cycle. Note that at this stage all pendant paths belong to H, and that we can also assume that the ribbon edge $\{u, v\}$ belongs to H.

(e) If $\hat{H}$ is a Hamiltonian cycle, we are done. Otherwise, $\hat{H}$ is a spanning path; see Fig. 1(e) for an example with $d = 4$. Let y_1 and y_2 be the two leaves of $\hat{H}$, and let w_1 and w_2 be the two original vertices of $\bar{G}$ which are closest in $\hat{H}$ to y_1 and y_2, respectively (again, it may happen that $w_1 = y_1$ or $w_2 = y_2$). We remove from H the paths from y_1 to w_1 and from y_2 to w_2, and add the direct path from w_1 to w_2; see Fig. 1(f). After this transformation, $\hat{H}$ is a Hamiltonian cycle, and $|E(G)|$ (resp. $w(H)$) has decreased by at most 4 (resp. 3).

After these transformations, which can clearly be done in polynomial time, we have obtained from the initial solution H another solution H_1 such that $\hat{H}_1$ is a Hamiltonian cycle and such that all pendant paths of $\bar{G}$, as well as the ribbon edge $\{u, v\}$, belong to H_1. In particular, in $\hat{H}$ all twin contact edges come in pairs. In the above transformations, the only step in which $|E(H)|$ or $w(H)$ may have decreased is the last one. Therefore, $|E(H_1)| \geq |E(H)| - 4$ and $w(H_1) \geq w(H) - 3$.

In order to obtain S_1 from H_1, we do the following. Let P be a piece of G^{2^p} for which $|E(S) \cap E(P)|$ is maximized. We just replace each white vertex in H_1 with the piece P. By construction, S_1 is a solution of MDBCS$_d$ in G^{2^p} in which all twin 1-contact edges come in pairs. Let us now argue about $|E(S_1)|$ with respect to $|E(S)|$. Let $x = |E(S) \cap E(P)|$, and note that $w(H)$ is the number of pieces of G^{2^p} with non-empty intersection with S. For this analysis, let n be the number of original vertices in $\bar{G}$. From the above definitions and by construction, it holds that $|E(S)| \leq w(H)(x + 2)$ and that $|E(S_1)| \geq w(H_1)(x + 2)$, so $|E(S_1)|/|E(S)| \geq w(H_1)/w(H)$. As $\hat{H}_1$ is a Hamiltonian cycle and all pendant paths of $\bar{G}$ belong to H_1, we have that $w(H_1) \geq (d - 2 + 1)n = (d - 1)n$, and using that $w(H_1) \geq w(H) - 3$, we get that

$$w(H_1) \geq \max\{\, (d-1)n,\ w(H) - 3 \,\}. \tag{7}$$

We distinguish two cases according to $w(H)$. First, if $w(H) < n$, then using Inequality (7) we get

$$\frac{|E(S_1)|}{|E(S)|} \geq \frac{(d-1)n}{w(H)} > \frac{(d-1)n}{n} = d - 1.$$

In particular, in this case we have that $|E(S_1)| > |E(S)|$. Otherwise, if $w(H) \geq n$, using again Inequality (7) we get

$$\frac{|E(S_1)|}{|E(S)|} \geq \frac{w(H) - 3}{w(H)} \geq 1 - \frac{3}{n},$$

and therefore

$$|E(S_1)| \geq |E(S)| - \frac{3|E(S)|}{n} = \ell - o(\ell),$$

as we wanted to prove.

Let us now explain how the above procedure is iterated for $i = 2, \ldots, p$. We recursively repeat the transformations describe above in each piece of G^{2^p} intersected by the current solution, sequentially modifying it to another one with the desired property. Note that after each step, all pieces and the intersection of the current solution with them are identical, so only one transformation is required for each $i = 2, \ldots, p$, and in all pieces the current solution gets replaced with the same new one. More precisely, in step i the *pieces* are naturally defined as the smallest connected components when removing pairs of twin i-contact edges. Note that as we can assume that for each such piece the ribbon edge between its contact vertices belongs to the solution, the intersection of the current solution with each piece is a connected subgraph of maximum degree at most d, and hence a valid solution of MDBCS$_d$. Therefore, the arguments above can be safely repeated recursively for $i = 2, \ldots, p$, and in each step i we obtain a solution in which all twin j-contact edges with $j \leq i$ come in pairs. At each step, the number of edges of the solution in each piece is reduced only by a lower order additive factor, and therefore for $i = p$ we obtain a solution S' with $\ell - o(\ell)$ edges such that a contact edge belongs to S' if and only if its twin contact edge does, as claimed. Clearly, the overall running time is polynomial in the number of vertices of G^{2^p}. □

We are now ready to prove the following lemma, which is the main ingredient of the proof of Theorem 3.2.

Lemma 3.3. *Let $d \geq 3$ and $p \geq 1$ be two fixed integers, and let $G \in \mathcal{G}$. Given a solution S of* MDBCS$_d$ *in G^{2^p} with ℓ edges, one can find in polynomial time a solution H of* MDBCS$_d$ *in $G^{2^{p-1}}$ with $\sqrt{\ell} - o(\sqrt{\ell})$ edges.*

Proof. We will construct from S the desired solution H in $G^{2^{p-1}}$. First, we apply Lemma 3.2 and obtain from S in polynomial time a solution S' in G^{2^p} with $\ell - o(\ell)$ edges such that a contact edge belongs to S' if and only if its twin contact edge does. Observe that a subgraph of G^{2^p} can pass from one copy of $G^{2^{p-1}}$ corresponding to an edge to another copy of $G^{2^{p-1}}$

corresponding to another edge only via contact vertices and edges. Let $\ell' = |E(S')|$, and note that $\ell' = \ell - o(\ell)$. In order to define H, we distinguish two cases:

(i) S' intersects strictly fewer than $\sqrt{\ell'}$ copies of $G^{2^{p-1}}$ in G^{2^p}.
Then let $H = S' \cap G_e^{2^{p-1}}$, with $G_e^{2^{p-1}}$ being a copy of $G^{2^{p-1}}$ in G^{2^p} such that $|E(S') \cap E(G_e^{2^{p-1}})|$ is maximized. As by Lemma 3.2 we can assume that the ribbon edge of the copy $G_e^{2^{p-1}}$ belongs to S', it follows that H is connected. It is also clear that, as $\Delta_{S'} \leq d$, then $\Delta_H \leq d$ as well. Let us now argue about $|E(H)|$.
Suppose that S' intersects c copies of $G^{2^{p-1}}$, with $c < \sqrt{\ell'}$, and for $1 \leq i \leq c$ let e_i be the number of edges of S' in the i-th copy of $G^{2^{p-1}}$. Note that by definition $|E(H)| = \max_{1 \leq i \leq c} e_i$. As by Lemma 3.2, for each copy of $G^{2^{p-1}}$ intersected by S' both contact edges belong to S', it holds

$$\sum_{i=1}^{c} (e_i + 2) \geq \ell'. \tag{8}$$

Since by assumption $c < \sqrt{\ell'}$, it follows from Inequality (8) that $\sum_{i=1}^{c} e_i > \ell' - 2\sqrt{\ell'}$, which in turn implies that

$$|E(H)| = \max_{1 \leq i \leq c} e_i \geq \frac{\ell' - 2\sqrt{\ell'}}{\sqrt{\ell'}} = \sqrt{\ell'} - 2 = \sqrt{\ell} - o(\sqrt{\ell}),$$

as we wanted to prove.

(ii) H intersects at least $\sqrt{\ell'}$ copies of $G^{2^{p-1}}$ in G^{2^p}.
By Lemma 3.2, we know that whenever S' intersects a copy of $G^{2^{p-1}}$, both twin contact edges of this copy belong to S'. We define H as the subgraph of $G^{2^{p-1}}$ induced by the edges $e \in E(G^{2^{p-1}})$ such that $E(S') \cap E(G_e^{2^{p-1}}) \neq \emptyset$, that is, the edges of $G^{2^{p-1}}$ whose corresponding copy has a non-empty intersection with S', plus the contact and ribbon edges of G^{2^p} belonging to S' which do not correspond to a copy of $G^{2^{p-1}}$. This subgraph H is clearly connected by the connectivity of S', and it holds that $\Delta_H \leq d$ because $\Delta_{S'} \leq d$ and in S' all twin contact edges come in pairs. Finally, $|E(H)|$ is bounded below by the number of intersected copies of $G^{2^{p-1}}$, which is at least $\sqrt{\ell'} = \sqrt{\ell} - o(\sqrt{\ell})$, as we wanted to prove.

Since the above operations can clearly be performed in polynomial time and either case (i) or (ii) must necessarily occur, the lemma follows. □

Using Lemmas 3.1 and 3.3 one can show the following theorem, inspired from [24, Theorem 8].

Theorem 3.2. *If for some fixed integer $d \geq 3$, MDBCS$_d$ has a polynomial-time algorithm that achieves a constant-factor approximation in the class of graphs $\mathcal{G}^*$, then it has a* PTAS *in the class of graphs $\mathcal{G}$.*

Proof. Let $\mathcal{A}$ be an algorithm that achieves a C-approximation for MDBCS$_d$ in the class of graphs $\mathcal{G}^*$, for some fixed constant $C > 1$. We restrict the input graphs to belong to the class of graphs $\mathcal{G}$. Given an input graph $G = (V, E)$ belonging to $\mathcal{G}$, we build the graph G^{2^p} by applying p times the edge squaring operation, where p is an integer to be specified later. For $k = 0, \ldots, p$, let OPT$_k$ be the number of edges of an optimal solution of MDBCS$_d$ in G^{2^k}, and let for simplicity OPT $=$ OPT$_0$. By Lemma 3.1, for $p \geq 1$ it holds

$$\mathrm{OPT}_p \geq \mathrm{OPT}^{2^p} - o(\mathrm{OPT}^{2^p}). \tag{9}$$

The PTAS is now obtained as follows. We run algorithm $\mathcal{A}$ on the graph G^{2^p}, yielding a solution with at least OPT$_p/C$ edges, since $\mathcal{A}$ is a C-approximation algorithm. Beginning from this solution, we apply Lemma 3.3 p times to obtain a solution of MDBCS$_d$ in G with weight SOL, such that

$$\begin{aligned}
\mathrm{SOL} &\geq \left(\frac{\mathrm{OPT}_p}{C}\right)^{1/2^p} - o\left(\left(\frac{\mathrm{OPT}_p}{C}\right)^{1/2^p}\right) \\
&\geq \frac{\left(\mathrm{OPT}^{2^p} - o(\mathrm{OPT}^{2^p})\right)^{1/2^p}}{C^{1/2^p}} - o\left(\left(\frac{\mathrm{OPT}_p}{C}\right)^{1/2^p}\right) \\
&= \frac{\mathrm{OPT}}{C^{1/2^p}} - o(\mathrm{OPT}),
\end{aligned} \tag{10}$$

where we have used Eq. (9) in the second inequality, and in the last equality the fact that, by the definition of the edge squaring operation, $\mathrm{OPT}_p = \mathcal{O}(\mathrm{OPT}^{2^p})$, so $o\left(\left(\frac{\mathrm{OPT}_p}{C}\right)^{1/2^p}\right) = o(\mathrm{OPT})$. It is then clear from Eq. (10) that for any $\varepsilon > 0$, there exists an integer $p(\varepsilon, C)$ such that for any graph $G \in \mathcal{G}$ with $n = |V(G)|$ large enough, $\frac{\mathrm{OPT}}{\mathrm{SOL}} \leq 1 + \varepsilon$. Since for any fixed $\varepsilon > 0$,

the overall running time of this algorithm is polynomial in n, we have constructed a polynomial-time $(1+\varepsilon)$-approximation algorithm for any $\varepsilon > 0$ in the class of graphs $\mathcal{G}$. In other words, MDBCS$_d$ admits a PTAS for any fixed integer $d \geq 3$ in the class of graphs $\mathcal{G}$, as claimed. □

Corollary 3.2 and Theorem 3.2 together yield the following theorem.

Theorem 3.3. *The* MDBCS$_d$ *problem does not admit any constant-factor approximation algorithm for any fixed* $d \geq 3$*, even if the input graph is restricted to belong to the class of graphs* $\mathcal{G}^*$*, unless* P = NP.

Karger et al. ruled out in [24] the existence of weaker approximation algorithms for finding a longest path in a given graph. In the same spirit we show the following theorem.

Theorem 3.4. *If there is a polynomial-time algorithm for* MDBCS$_d$ *for some fixed integer* $d \geq 3$*, with approximation ratio* $2^{\mathcal{O}(\sqrt{\log n})}$*, then* NP $\subseteq$ DTIME$(2^{\mathcal{O}(\log^5 n)})$*. The result also holds if all edges have unitary weight.*

Proof. Let $\mathcal{A}$ be an algorithm of approximation ratio $g(n) = 2^{\mathcal{O}(\sqrt{\log n})}$ for MDBCS$_d$, in particular in the class of graphs $\mathcal{G}^*$. Let $G = (V, E) \in \mathcal{G}^*$ be an instance of MDBCS$_d$ with n vertices and having an optimal solution with ℓ edges. We choose p to be the smallest integer such that $N = n^{3^p} \geq 2^{\log^5 n}$. Now we generate from G the graph G^{2^p} by applying p times the edge squaring operation. Note that the number of vertices of G^{2^p} is bounded above by N. By Lemma 3.1, we know that G^{2^p} has a solution with at least $\ell^{2^p} - o(\ell^{2^p})$ edges. Running $\mathcal{A}$ on G^{2^p} we obtain a solution H with at least $\left(\ell^{2^p} - o(\ell^{2^p})\right)/g(N)$ edges. Furthermore, starting with the solution H and repeatedly applying Lemma 3.3, we obtain a solution of MDBCS$_d$ in G with at least $(\ell - o(\ell))/h(n) - o(\ell)$ edges, where

$$h(n) = g(N)^{1/2^p} = 2^{\mathcal{O}(\sqrt{\log N}/2^p)} = 2^{\mathcal{O}(\log^{2.5} n/2^p)} = \mathcal{O}(1),$$

where the last equality follows because $n^{3^p} \geq 2^{\log^5 n}$ implies that $\log^{5/2} n \leq (3^{5/8})^p < 2^p$.

This implies that we can approximate MDBCS$_d$ in the class of graphs $\mathcal{G}^*$ within a constant factor in time polynomial in N, that is, in time $2^{\mathcal{O}(\log^5 n)}$. But by Theorem 3.3 we know that finding a constant-factor approximation for MDBCS$_d$ in the class of graphs $\mathcal{G}^*$ is NP-hard, hence we have given a simulation of an NP-hard problem in time $2^{\mathcal{O}(\log^5 n)}$. The theorem follows. □

4. Approximating MDBCS$_d$

In this section we focus on approximating MDBCS$_d$. As we have seen in Section 3, MDBCS$_d$ does not admit any constant-factor approximation algorithm in general graphs.

First, we provide in Section 4.1 approximation algorithms in general graphs for both the weighted and unweighted versions of the problem. Then, we show in Section 4.2 that when the input graph has a low-degree spanning tree (in terms of d), the problem becomes easy to approximate in weighted and unweighted graphs. Specifically, Proposition 4.2 provides a constant-factor approximation for such graphs.

4.1. General graphs

The first non-trivial approximation algorithm for the Longest Path problem (which corresponds to the case $d = 2$ of MDBCS$_d$, as discussed in the introduction) has approximation ratio $\mathcal{O}(n/\log n)$ [3] (see Section 8 for a discussion about the recent advances on the Longest Path problem). Using the results of [3], we provide in Theorem 4.2 an approximation algorithm for MDBCS$_d$ in general unweighted graphs for any $d \geq 2$. We then turn to weighted graphs and provide an approximation algorithm in Theorem 4.3. Finally we compare both algorithms for unweighted graphs. These are the first approximation algorithms for MDBCS$_d$ in general graphs for $d \geq 3$.

We need a preliminary lemma, that uses the following result.

Proposition 4.1 (*Munro and Raman [29]*)**.** *Any unordered tree on n nodes can be represented using* $2n + o(n)$ *bits with adjacency being supported in* $\mathcal{O}(n)$ *time.*

Let $\mathcal{T}_{n,d}$ be the set of non-isomorphic unlabeled trees on n nodes with maximum degree at most d.

Lemma 4.1. *The set* $\mathcal{T}_{\log n,d}$ *can be generated in polynomial time in n.*

Proof. It is well known that $|\mathcal{T}_{n,n-1}| \sim C\alpha^n n^{-5/2}$ as $n \to \infty$, for positive constants C and α, cf. [33]. Hence, the set $\mathcal{T}_{\log n,\log n-1}$ is of size polynomial in n. In addition, one can efficiently generate all the elements of $\mathcal{T}_{\log n,\log n-1}$. Indeed by Proposition 4.1 any unlabeled tree on $\log n$ nodes can be represented using $2\log n + o(\log n)$ bits with adjacency being supported in $\mathcal{O}(\log n)$ time. Finally, the set $\mathcal{T}_{\log n,d}$ is obtained from $\mathcal{T}_{\log n,\log n-1}$ by removing all the elements T with $\Delta_T > d$, where Δ_T is the maximum degree of the tree T. □

The main ingredient of our first algorithm is the following theorem of Alon et al. [3], which is based on the *color-coding* method.

Theorem 4.1 (*Alon et al. [3]*). *If a graph $G = (V, E)$ contains a subgraph isomorphic to a graph $H = (V_H, E_H)$ whose tree-width is at most t, then such a subgraph can be found in $2^{\mathcal{O}(|V_H|)} \cdot |V|^{t+1} \cdot \log |V|$ time.*

In particular, trees on $\log |V|$ vertices can be found in time $|V|^{\mathcal{O}(1)} \cdot \log |V|$. We are ready to describe our algorithm for unweighted graphs.

Algorithm $\mathcal{A}$:

(1) Generate all the elements of $\mathcal{T}_{\log n,d}$. Define the set $\mathcal{F} \leftarrow \emptyset$.
(2) For each $T \in \mathcal{T}_{\log n,d}$, test if G contains a subgraph isomorphic to T. If such a subgraph is found, add it to $\mathcal{F}$.
(3) If $\mathcal{F} = \emptyset$ OR $d > \log n$, output an arbitrary connected subgraph of G with d edges. Otherwise, output any element in $\mathcal{F}$.

Theorem 4.2. *For all $d \geq 2$, algorithm $\mathcal{A}$ provides a ρ-approximation algorithm for* MDBCS$_d$ *in unweighted graphs, with $\rho = \min\{m, nd/2\}/\log n$.*

Proof. Let us first observe that the running time of algorithm $\mathcal{A}$ is polynomial in n. Indeed, steps (1) and (2) can be executed in polynomial time by Lemma 4.1 and Theorem 4.1, respectively. Step (3) takes time proportional to the size of the output. Algorithm $\mathcal{A}$ is clearly correct, since by definition of the set $\mathcal{T}_{\log n,d}$ the output graph is a solution of MDBCS$_d$ in G.

Finally, let us consider the approximation ratio of algorithm $\mathcal{A}$. Let OPT be the number of edges of an optimal solution of MDBCS$_d$ in G, and let ALG be the number of edges of the solution found by algorithm $\mathcal{A}$. We distinguish two cases:

- If $\text{OPT} \geq \frac{d \cdot \log n}{2}$, then any optimal solution $\hat{H}$ has at least $\log n$ vertices. In particular, $\hat{H}$ contains a tree on $\log n$ vertices with maximum degree at most d, and so does G. Hence, this tree will be found in step (2), and therefore $\text{ALG} \geq \log n - 1$. (In the sequel we assume for the sake of simplicity that $\text{ALG} \geq \log n$.) On the other hand, we know that $\text{OPT} \leq \min\{m, nd/2\}$.
- Otherwise, if $\text{OPT} < \frac{d \cdot \log n}{2}$, then we use the fact that $\text{ALG} \geq d$. Note that such a connected subgraph with d edges can be greedily found starting from any node of G.

In both cases,

$$\frac{\text{OPT}}{\text{ALG}} \leq \max\left\{\frac{\min\left\{m, \frac{nd}{2}\right\}}{\log n}, \frac{\log n}{2}\right\} = \frac{\min\{m, nd/2\}}{\log n}, \quad \text{since } \log n = \mathcal{O}(\sqrt{n}).$$

The theorem follows. □

In particular, for $d = 2$ Algorithm $\mathcal{A}$ is equivalent to the LONGEST PATH algorithm of [3].

Theorem 4.3. *The* MDBCS $_d$ *problem admits a ρ-approximation algorithm $\mathcal{B}$ in weighted graphs, with $\rho = \min\{n/2, m/d\}$.*

Proof. Let us describe algorithm $\mathcal{B}$. Let F be the set of d heaviest edges in the input graph G, and let W be the set of endpoints of those edges. We distinguish two cases according to the connectivity of the subgraph $H = (W, F)$. Let $\omega(F)$ denote the total weight of the edges in F.

If H is connected, the algorithm returns H. We claim that this yields a ρ-approximation. Indeed, if an optimal solution consists of m^* edges of total weight ω^*, then $\text{ALG} = \omega(F) \geq \frac{\omega^*}{m^*} \cdot d$, since by the choice of F the average weight of the edges in F cannot be smaller than the average weight of the edges of an optimal solution. As $m^* \leq m$ and $m^* \leq dn/2$, we get that $\text{ALG} \geq \frac{\omega^*}{m} \cdot d$ and $\text{ALG} \geq \frac{\omega^*}{dn/2} \cdot d = \frac{\omega^*}{n/2}$.

Now suppose $H = (W, F)$ consists of a collection $\mathcal{F}$ of k connected components. Then we *glue* these components together in $k - 1$ phases. In each phase, we pick two components $C, C' \in \mathcal{F}$, and combine them into a new connected component $\hat{C}$ by adding a connecting path, without touching any other connected component of $\mathcal{F}$. We then set $\mathcal{F} \leftarrow \mathcal{F} \setminus \{C, C'\} \cup \{\hat{C}\}$.

Each phase operates as follows. For every two components $C, C' \in \mathcal{F}$, compute their distance, defined as $d(C, C') = \min\{\text{dist}(u, u', G) \mid u \in C, u' \in C'\}$, where $\text{dist}(u, u', G)$ is the length of a shortest path in G between u and u'. Take a pair $C, C' \in \mathcal{F}$ attaining the smallest distance $d(C, C')$. Let $u \in C$ and $u' \in C'$ be two vertices realizing this distance, i.e., such that $\text{dist}(u, u', G) = d(C, C')$. Let $p(u, u')$ be a shortest path between u and u' in G. Let $\hat{C}$ be the connected component obtained by merging C, C' and the path $p(u, u')$.

For the correctness proof, we need the following two observations: First, observe that in every phase, the path $p(u, u')$ used to merge the components C and C' does not go through any other cluster C'', since otherwise, $d(C, C'')$ would be strictly smaller than $d(C, C')$, contradicting the choice of the pair (C, C'). Moreover, $p(u, u')$ does not go through any other vertex v in the cluster C except for its endpoint u, since otherwise, $\text{dist}(v, u', G) < \text{dist}(u, u', G)$, contradicting the choice of the pair u, u'. Similarly, $p(u, u')$ does not go through any other vertex v' in C'.

We now claim that after i phases, the maximum degree of H satisfies $\Delta_H \leq d - k + i + 1$. This is proved by induction on i. For $i = 0$, i.e., for the initial graph $H = (W, F)$, we observe that as F consists of d edges arranged in k separate components, the largest component will have no more than $d - k + 1$ edges, hence $\Delta_H \leq d - k + 1$, as required. Now suppose the claim holds after $i - 1$ phases, and consider phase i. All nodes other than those of the path $p(u, u')$ maintain their degree from the previous phase. The nodes u and u' increase their degree by 1, so by the inductive hypothesis, their new degree is at most $(d - k + (i - 1) + 1) + 1 = d - k + i + 1$, as required. Finally, the intermediate nodes of $p(u, u')$ have degree $2 \leq d - k + i + 1$

(since $i \geq 1$ and $k \leq d$). It follows that by the end of phase $k-1$, $\Delta_H \leq d-k+k-1+1 = d$. Also, at that point H is connected. Hence H is a valid solution.

Finally, the approximation ratio of the algorithm is still at most $\rho = \min\{n/2, m/d\}$, since this ratio was guaranteed for the originally selected F, and the final subgraph contains the set F. □

For unweighted graphs, comparing approximation ratios of Algorithm $\mathcal{A}$ of Theorem 4.2 and Algorithm $\mathcal{B}$ of Theorem 4.3, we conclude that Algorithm $\mathcal{A}$ performs better when $d < \log n$, while Algorithm $\mathcal{B}$ is better when $d \geq \log n$. So if we run both the algorithms and keep the best solution, we obtain the following corollary.

Corollary 4.1. *In unweighted graphs, the* MDBCS $_d$ *problem admits a ρ-approximation algorithm, with* $\rho = \min\{n/2,\ nd/(2\log n),\ m/d,\ m/\log n\}$.

4.2. Graphs with low-degree spanning trees

We first state a simple lemma about the optimal solutions of the polynomially solvable MDBS$_d$ problem, whose definition is the same as the MDBCS$_d$ problem, except that the connectivity of the output subgraph is not required.

Lemma 4.2. *Given a graph G, an integer $d \geq 2$, and a real number k with $1 < k \leq d$, let* OPT$_d$ *and* OPT$_{d/k}$ *be the optimal solutions of* MDBS$_d$ *and* MDBS$_{\lceil d/k\rceil}$ *in G, respectively. Then* $\mathrm{OPT}_d \leq \frac{d+1}{d} \cdot k \cdot \mathrm{OPT}_{d/k}$.

Proof. Let $\hat{H}_d$ be a subgraph of G attaining OPT$_d$. By Vizing's theorem [14], there exists a coloring of the edges of $\hat{H}_d$ using at most $d+1$ colors. Order these chromatic classes according to non-increasing total edge-weight, and let $H_{d/k}$ be the subgraph of G induced by the first $\lceil d/k\rceil$ classes. Then the maximum degree of $H_{d/k}$ does not exceed $\lceil d/k\rceil$, and the sum of its edge weights is at least $\frac{d\cdot \mathrm{OPT}_d}{k\cdot(d+1)}$. Hence

$$\mathrm{OPT}_d \leq \frac{d+1}{d} \cdot k \cdot \mathrm{OPT}_{d/k}. \quad \square$$

Note that, in particular, $\mathrm{OPT}_d \leq \frac{3k}{2} \cdot \mathrm{OPT}_{d/k}$. For example, if $G = C_5$ and $d = k = 2$, then $\mathrm{OPT}_2 = 5 \leq 3/2 \cdot 2 \cdot \mathrm{OPT}_1 = 3 \cdot 2 = 6$.

We define a *k-tree* of a connected graph to be a spanning tree with maximum degree at most k. We are now ready to describe our approximation algorithm.

Proposition 4.2. *Given an integer $d \geq 2$ and a real number ℓ with $1 < \ell < d$, let $\mathcal{G}_{d,\ell}$ be the class of graphs that have a $(d/\ell - 1)$-tree. Then, for any $G \in \mathcal{G}_{d,\ell}$,* MDBCS$_d$ *can be approximated in G within a constant factor* $\frac{d+1}{d}\frac{\ell}{\ell-1}$.

Proof. Since G has a $(d/\ell-1)$-tree, by [19] one can find in polynomial time a spanning tree T of G with maximum degree at most d/ℓ. Let $k = \frac{\ell}{\ell-1}$, and let H be the optimal solution of MDBS$_{\lceil d/k\rceil}$ in G (recall that MDBS$_d$ is in P, but the output graph is not necessarily connected). Then the graph $T \cup H$ is a solution of MDBCS$_d$ in G, since it is connected and has maximum degree at most d. By Lemma 4.2 and using the fact that any solution of MDBCS$_d$ is also a solution of MDBS$_d$, we conclude that $T \cup H$ provides a $\frac{d+1}{d}\frac{\ell}{\ell-1}$-approximation for MDBCS$_d$ in G. □

For example, Proposition 4.2 states that MDBCS$_d$ admits a $\left(2 \cdot \frac{d+1}{d}\right)$-approximation in graphs with a spanning tree of maximum degree at most $d/2-1$. Note that $2 \cdot \frac{d+1}{d} \leq 8/3$ for any $d \geq 3$.

4.2.1. The relation between MDBCS$_d$ and graph toughness

Given a graph G, denote by $\kappa(G)$ the number of connected components of G.

Definition 4.1 (*Toughness of a Graph [37]*)**.** The *toughness* $t(G)$ of a graph $G = (V, E)$ is the largest number t such that, for any subset $S \subseteq V$, $|S| \geq t \cdot \kappa(G[V \setminus S])$, provided that $\kappa(G[V \setminus S]) > 1$.

It is proved in [37] that if $t(G) \geq \frac{1}{k-2}$, for $k \geq 3$, then G has a k-tree.

Theorem 4.4 (*Win [37]*)**.** *Let G be a graph. If $t(G) \geq \frac{1}{k-2}$, with $k \geq 3$, then G has a k-tree.*

Let us relate the above definitions with the MDBCS$_d$ problem. If a graph G does not satisfy the conditions of Proposition 4.2, then G does not have a $(d/2-1)$-tree. In this case one has some additional knowledge about the structure of G. Namely, Theorem 4.4 states that, provided that $d \geq 8$, the *toughness* $t(G)$ of G satisfies $t(G) < \frac{1}{d/2-3}$, implying that there exists a subset $S \subseteq V(G)$ such that

$$\kappa(G[V \setminus S]) > |S| \cdot \left(\frac{d}{2} - 3\right).$$

It would be interesting to explore the question whether this structural result permits to approximate MDBCS$_d$ efficiently.

5. Hardness of approximating MSMD_d

The main result of this section, Theorem 5.4, states that MSMD_d does not admit a constant-factor approximation in general graphs, for $d \geq 3$. We first prove in Section 5.1 that MSMD_d does not admit a PTAS, and then use the error amplification technique to prove the main result. Our reduction is obtained from the Vertex Cover (VC) problem (see Section 2).

5.1. MSMD_d does not admit a PTAS for any $d \geq 3$

The result is first established in Theorem 5.1 for the case $d = 3$. An easy extension of Theorem 5.1 allows to prove the result for any $d \geq 3$ in Theorem 5.2. For the sake of completeness, the proof of Theorem 5.2 can be found in Appendix A.

Theorem 5.1. *The* MSMD_3 *problem does not admit a* PTAS, *unless* P = NP.

Proof. We present a reduction from Vertex Cover, which does not admit a PTAS in cubic graphs, unless P = NP [2]. Given a cubic graph H as instance of Vertex Cover, with $|V(H)| = n$, we construct an instance $G = f(H)$ of MSMD_3 as follows. Without loss of generality, we may assume that $|E(H)| = 3n/2 = 3 \cdot 2^{\ell}$ for some integer ℓ. Let T be the rooted tree with root r and height $\ell + 1$ on $3 \cdot 2^{\ell+1} - 2$ vertices, in which all the internal vertices have degree three (thus, containing $3 \cdot 2^{\ell}$ leaves). We identify the leaves of T with the elements in $E(H)$, and denote –with slight abuse of notation–this set by E (note that $E \subseteq V(T)$). We add another copy of E, called F, and a Hamiltonian cycle on $E \cup F$ inducing a bipartite graph with partition classes E and F, as shown in Fig. 2. We also identify the vertices of F with the elements in $E(H)$. Now we add a set A of $|V(H)|$ new vertices identified with the elements in $V(H)$, and join them to the vertices in F according to the incidence relations in H: we add an edge between a vertex in F corresponding to $e \in E(H)$ and a vertex in A corresponding to $u \in V(H)$ if and only if e contains u. This completes the construction of G, which is illustrated in Fig. 2.

We now claim that minimum subgraphs of G of minimum degree at least 3 correspond to minimum vertex covers of H, and vice-versa. To see this, first note that if such a subgraph D of G contains a vertex of $V(T) \cup F$, then it should contain all the vertices of $V(T) \cup F$, because of the construction of G and the degree constraints. On the other hand, D cannot contain only vertices of A (as they induce an independent set), hence D must contain all the vertices of $V(T) \cup F$. Note that all the vertices of F have degree two in $G[V(T) \cup F]$. Therefore, the problem reduces to finding a smallest subset of vertices in A covering all the vertices in F. This is exactly the Vertex Cover problem in H. Thus, we have that

$$\text{OPT}_{\text{MSMD}_3}(G) = \text{OPT}_{\text{VC}}(H) + |V(T)| + |F| = \text{OPT}_{\text{VC}}(H) + 9n/2 - 2. \tag{11}$$

(We will omit in the sequel the reference to G and H in $\text{OPT}_{\text{MSMD}_3}$ and OPT_{VC}, respectively.) Note also that any solution of MSMD_3 in G of size $\text{SOL}_{\text{MSMD}_3}$ defines a solution of Vertex Cover in H of size $\text{SOL}_{\text{VC}} = \text{SOL}_{\text{MSMD}_3} - 9n/2 + 2$. Assume now for contradiction that MSMD_3 admits a PTAS, that is, for any $\varepsilon > 0$ we can find in polynomial time a solution of MSMD_3 in G of size $\text{SOL}_{\text{MSMD}_3} \leq (1 + \varepsilon) \cdot \text{OPT}_{\text{MSMD}_3}$. Therefore, we could find in polynomial time a solution of Vertex Cover in H of size

$$\text{SOL}_{\text{VC}} = \text{SOL}_{\text{MSMD}_3} - 9n/2 + 2 \leq (1 + \varepsilon) \cdot \text{OPT}_{\text{MSMD}_3} - 9n/2 + 2. \tag{12}$$

Using Eq. (11) in Eq. (12) we get

$$\text{SOL}_{\text{VC}} \leq (1 + \varepsilon) \cdot \text{OPT}_{\text{VC}} + \varepsilon \cdot (9n/2 - 2). \tag{13}$$

Note that since H is cubic, any vertex cover of H has size at least $|E(H)|/3 = n/2$, so in particular $n/2 \leq \text{OPT}_{\text{VC}}$. Using this inequality in Eq. (13) yields

$$\text{SOL}_{\text{VC}} \leq (1 + \varepsilon) \cdot \text{OPT}_{\text{VC}} + \varepsilon \cdot (9 \cdot \text{OPT}_{\text{VC}} - 2) \leq (1 + 10\varepsilon) \cdot \text{OPT}_{\text{VC}}.$$

Therefore, the existence of a PTAS for MSMD_3 would imply the existence of a PTAS for Vertex Cover in cubic graphs, which is impossible unless P = NP [2]. □

Theorem 5.2. *The* MSMD_d *problem does not admit a* PTAS *for any fixed* $d \geq 3$, *unless* P = NP.

5.2. MSMD_d is not in APX for any $d \geq 3$

We are now ready to prove the main result of this section. Again, we focus on the case $d = 3$ in Theorem 5.3 and then extend the ideas for any $d \geq 3$ in Theorem 5.4, whose proof can be found in Appendix B.

Theorem 5.3. *The* MSMD_3 *problem does not admit any constant-factor approximation, unless* P = NP.

Proof. The proof is by appropriately applying the standard error amplification technique. Let $\mathcal{G}_1 = \{G\}$ be the family of graphs constructed in Theorem 5.1 (see Fig. 2) from the instances H of Vertex Cover, G being a typical member of this family, and let $\alpha > 1$ be the factor of inapproximability of MSMD_3, that exists by Theorem 5.1.

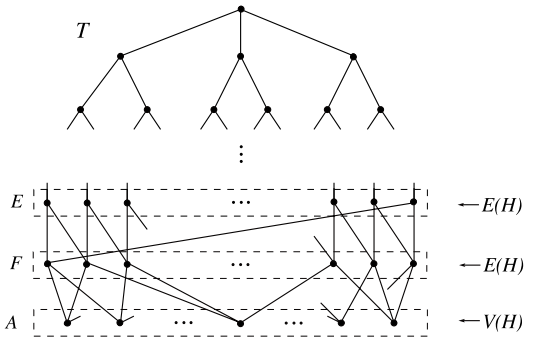

Fig. 2. An example of the graph G built in the reduction of Theorem 5.1.

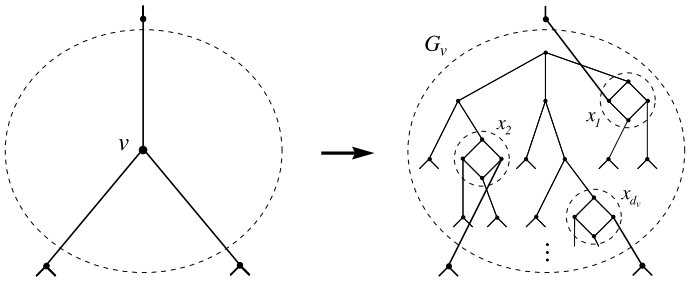

Fig. 3. Error amplification in the proof of Theorem 5.3.

We construct a sequence of families of graphs $\mathcal{G}_k$, such that MSMD_3 is hard to approximate within a factor $\theta(\alpha^k)$ in the family $\mathcal{G}_k$. This proves that MSMD_3 does not have any constant-factor approximation. In the following, G_k will denote a typical element of $\mathcal{G}_k$ constructed from the element $G \in \mathcal{G}_1$. We describe the construction of G_2, and obtain the result by repeating the same construction inductively to obtain G_k. For every vertex v in G, we construct a graph G_v as follows. First, letting $d_v = \deg_G(v)$, take a copy of G and choose d_v other arbitrary vertices $x_1, \ldots, x_{d_v}$ of degree three in $T \subset G$. Then, replace each of these vertices x_i with a cycle of length four, and join three of the vertices of the cycle to the three neighbors of x_i, $i = 1, \ldots, d_v$. Let G_v be the graph obtained in this way. Note that G_v contains exactly d_v vertices of degree two.

Now take a copy of G, and replace each vertex v with G_v. Then, join the d_v edges incident to v to the d_v vertices of degree two in G_v. This completes the construction of the graph G_2, which is illustrated in Fig. 3.

We have that $|V(G_2)| = |V(G)|^2 + o(|V(G)|^2)$, because each vertex of G is replaced with a copy of G where we had replaced some of the vertices with a cycle of length four.

To find a solution of the MSMD_3 problem in G_2, note that for any $v \in V(G)$, once a vertex in G_v is chosen, we have to solve MSMD_3 in G, which is hard up to a constant factor α. But approximating the number of v's for which we should touch G_v is also solving MSMD_3 in G, which is hard up to the same factor α. This proves that approximating MSMD_3 in G_2 is hard up to a factor α^2. The proof of the theorem is completed by repeating this procedure, applying the same construction to obtain G_3, and inductively G_k. More precisely, in order to build G_k from G_{k-1}, we replace each vertex $v \in V(G_{k-1})$ with a copy of G_{k-1} in which $\deg_{G_{k-1}}(v)$ arbitrary vertices of degree three have been replaced with a cycle of length four. □

Theorem 5.4. *The* MSMD_d *problem does not admit any constant-factor approximation for any fixed* $d \geq 3$, *unless* $\text{P} = \text{NP}$.

6. Approximating MSMD_d in graphs with excluded minobelrs

Approximating MSMD_d seems to be really hard. Indeed, in contrast to many other problems for which there exist good approximation algorithms for restricted classes of graphs, like planar graphs or minor-free graphs, even obtaining an approximation algorithm for MSMD_d restricted to minor-free graphs appears to be challenging. These general frameworks and meta-theorems developed during the last years to obtain PTAS or constant-factor approximation algorithms for restricted classes of graphs (e.g., [12,13]) fail when applied to MSMD_d. For instance, the set of solutions of MSMD_d can be easily defined as a first-order logic formula, but it is unavoidable that the free set variable appears negatively, so the meta-theorem of Dawar et al. [12] cannot be applied. In this section, we briefly show how to obtain a simple $(n/\log n)$-approximation algorithm for the problem in minor-free graphs. We also stress that approximation algorithms for MSMD_d in general graphs are missing (see Section 8).

Our approximation algorithm for graphs with excluded minors has two main ingredients. The first one is that the MSMD_d problem can be solved in time single exponential on the tree-width of the input graph, and therefore the problem is in P

for graphs whose tree-width is $\mathcal{O}(\log n)$. This result is obtained using standard dynamic programming techniques, and can be found in [5]. More precisely, given a tree-decomposition of width t of an n-vertex graph, the time complexity to solve MSMD_d is $\mathcal{O}((d+1)^t(t+1)^{d^2}n)$.

The second ingredient is a recent powerful result of Demaine et al. [13]. They show that, for any fixed graph M, there is a constant c_M such that for every integer $k \geq 1$ and for every M-minor-free graph G, the vertices of G can be partitioned into $k+1$ sets such that any k of the sets induce a graph of tree-width at most $c_M k$. Furthermore, such a partition can be found in polynomial time.

By applying the latter result to an M-minor free graph G on n vertices for $k = \log n$, one can find a partition of $V(G)$ into $\log n + 1$ sets, such that the induced subgraph on any $\log n$ sets is of tree-width at most $c_M \log n$. The dynamic programming algorithm of [5] applied to the induced subgraph on any collection of $\log n$ sets permits to conclude whether G contains a subgraph of minimum degree at least d on at most $\log n$ vertices. This algorithm provides a polynomial-time $(n/\log n)$-approximation for MSMD_d in minor-free graphs.

7. Approximating DDD*k*S

In this section we provide approximation algorithms for the DDDkS problem. First we observe that the DDDkS problem is strongly related to the Dense k-Subgraph (DkS) problem, as stated in Proposition 7.1. The proof of this result is an easy exercise, and can be found for the sake of completeness in Appendix C.

Proposition 7.1. *The existence of an α-approximation algorithm for the* Dense k-Subgraph *problem implies the existence of a 2α-approximation algorithm for the* DDDkS *problem.*

For almost a decade, the best approximation ratio for the DkS problem has been $\mathcal{O}(n^\delta)$ for some universal constant $\delta < 1/3$, given by Feige et al. [18]. This algorithm has been very recently improved by Chlamtac and Feige [8], who provide an algorithm that for every $\varepsilon > 0$ approximates the DkS problem within a ratio of $n^{1/4+\varepsilon}$ in time $n^{\mathcal{O}(1/\varepsilon)}$. If allowed to run in time $n^{\mathcal{O}(\log n)}$, the algorithm achieves an approximation ratio of $\mathcal{O}(n^{1/4})$. According to Proposition 7.1, the same approximation ratios (modulo a factor 2) apply to DDDkS.

In the remainder of this section we provide a simple randomized $\mathcal{O}(\sqrt{n \log n})$-approximation algorithm[1] for the DDDkS problem, which does not use any "black-box" as subroutine (as it is the case of the algorithms following from Proposition 7.1).

Theorem 7.1. *The* DDDkS *problem admits a randomized $\mathcal{O}(\sqrt{n \log n})$-approximation algorithm.*

Proof. For every $1 \leq d \leq n$, let $H[d]$ be the maximum subgraph of G with minimum degree $\delta_{H[d]} \geq d$, in the sense that $H[d]$ contains any other subgraph H of G of minimum degree at least d. Also let $n[d] = |V(H[d])|$. The first stage of the algorithm computes $H[d]$ for every $1 \leq d \leq n$. This is easily done by initializing $H[1] = G$ and then successively removing from $H[d]$ all the vertices of degree at most d to obtain $H[d+1]$. Note that $n[d]$ can be zero, i.e., $H[d]$ can be the empty subgraph. The algorithm stops whenever it finds $n[d] = 0$.

Let $\tilde{d}$ be the index such that $n[\tilde{d}] > 0$ and $n[\tilde{d}+1] = 0$ (clearly $\tilde{d} \leq n-1$). If $k \geq n[\tilde{d}]$, then $H[\tilde{d}]$ is an exact solution of the problem, hence the output to the DkS problem is $\tilde{d}$. It remains to handle the case where $k < n[\tilde{d}]$. In this case, it is also clear that the solution d^* we are looking for is bounded by $\tilde{d}$, i.e., $d^* \leq \tilde{d}$. Two cases may occur.

- *Case a*: $k \leq 16\sqrt{n \log n}$ or $\tilde{d} \leq 16\sqrt{n \log n}$.
  In this case any connected subgraph of G of size at most k (for example a connected subtree of a spanning tree of G of size k, or even just an edge) has minimum degree at least one, hence it provides a solution that is within a factor $1/(16\sqrt{n \log n})$ of the optimal solution.
- *Case b*: Both $\tilde{d}, k > 16\sqrt{n \log n}$.
  Construct a subgraph H of $H[\tilde{d}]$ in the following way: select each vertex of $H[\tilde{d}]$ with probability $\sqrt{\log n}/\sqrt{n}$, and take H to be the induced subgraph of $H[\tilde{d}]$ by the set of selected vertices. Let $n_0 = |V(H)|$.

□

Claim 7.1. *The number of selected vertices satisfies $n_0 \leq 2n[\tilde{d}]\sqrt{\log n}/\sqrt{n}$ with probability at least $1 - 1/n^4$. In particular, $n_0 \leq k$ with probability at least $1 - 1/n^4$.*

Proof. Observe that n_0 can be expressed as the sum of $n[\tilde{d}]$ independent Boolean random variables $B_1, \ldots, B_{n[\tilde{d}]}$. Since $\mathbb{E}[n_0] = n[\tilde{d}]\sqrt{\log n}/\sqrt{n}$, applying Chernoff's bound on the upper tail yields

$$\mathbb{P}\text{rob}\left[B_1 + \cdots + B_{n[\tilde{d}]} > \frac{2n[\tilde{d}]\sqrt{\log n}}{\sqrt{n}}\right] < \exp\left(-\frac{n[\tilde{d}]\sqrt{\log n}}{4\sqrt{n}}\right).$$

[1] In the extended abstract presented in WAOA 2008 we provided an algorithm with ratio $\mathcal{O}(\sqrt{n}\log n)$. We thank an anonymous referee for suggesting us how to improve the algorithm to achieve the ratio $\mathcal{O}(\sqrt{n \log n})$.

Therefore, because $n[\tilde{d}] > k > 16\sqrt{n \log n}$, we have

$$\mathbb{P}\text{rob}\left[n_0 > \frac{2n[\tilde{d}]\sqrt{\log n}}{\sqrt{n}}\right] < \exp(-4 \log n) = \frac{1}{n^4},$$

and since $n[\tilde{d}] \leq n$, with probability at least $1 - \frac{1}{n^4}$, $n_0 \leq 2n[\tilde{d}]\sqrt{\log n}/\sqrt{n} \leq 2\sqrt{n}\sqrt{\log n} < 16\sqrt{n \log n} < k$. □

Claim 7.2. *For every vertex $v \in V(H)$, $\deg_H(v) \geq \frac{\tilde{d}\sqrt{\log n}}{2\sqrt{n}}$ with probability at least $1 - 1/n^2$.*

Proof. Observe first that $\deg_H(v)$ is a sum of $\deg_{H[\tilde{d}]}(v)$ independent Boolean random variables, and so the expected degree of v in H is $\deg_{H[\tilde{d}]}(v)\sqrt{\log n}/\sqrt{n} \geq \tilde{d}\sqrt{\log n}/\sqrt{n}$. This is because every vertex of $H[\tilde{d}]$ has degree at least $\tilde{d}$. This implies

$$\mathbb{P}\text{rob}\left[\deg_H(v) < \frac{\tilde{d}\sqrt{\log n}}{2\sqrt{n}}\right] \leq \mathbb{P}\text{rob}\left[\deg_H(v) < \frac{\deg_{H[\tilde{d}]}(v)\sqrt{\log n}}{2\sqrt{n}}\right].$$

Applying Chernoff's bound on the lower tail we have

$$\mathbb{P}\text{rob}\left[\deg_H(v) < \frac{\deg_{H[\tilde{d}]}(v)\sqrt{\log n}}{2\sqrt{n}}\right] < \exp\left(-\frac{\deg_{H[\tilde{d}]}(v)\sqrt{\log n}}{8\sqrt{n}}\right) \leq \exp\left(-\frac{\tilde{d}\sqrt{\log n}}{8\sqrt{n}}\right),$$

which in turn implies (because $\tilde{d} > 16\sqrt{n \log n}$),

$$\mathbb{P}\text{rob}\left[\deg_H(v) < \frac{\tilde{d}\sqrt{\log n}}{2\sqrt{n}}\right] \leq \exp\left(-\frac{16\sqrt{n}\log n}{8\sqrt{n}}\right) = \frac{1}{n^2}.$$ □

Claim 7.3. *$\delta_H \geq \tilde{d}\sqrt{\log n}/(2\sqrt{n})$ with probability at least $1 - 1/n$.*

Proof. By Claim 7.2, the probability that *any* node v of H has $\deg_H(v) < \tilde{d}\sqrt{\log n}/(2\sqrt{n})$ is at most $\frac{1}{n^2} \cdot |H| \leq 1/n$. □

Claims 7.1 and 7.3 together show that with probability at least $1 - \frac{1}{n} - \frac{1}{n^4} \geq 1 - \frac{2}{n}$, H has at most k vertices and has minimum degree at least $\tilde{d}\sqrt{\log n}/(2\sqrt{n})$. Therefore, with high probability, H provides a solution of DkS which is within a factor $\sqrt{\log n}/(2\sqrt{n})$ of the optimal solution. This concludes the proof of the theorem. □

8. Conclusions

This paper considered three degree-constrained subgraph problems and studied their behavior in terms of approximation algorithms and hardness of approximation. Our main results and several interesting questions that remain open are discussed below.

We proved that the MDBCS$_d$ problem is not in Apx for any $d \geq 2$, and that if there is a polynomial-time algorithm for MDBCS$_d$, $d \geq 2$, with an approximation ratio of $2^{\mathcal{O}(\sqrt{\log n})}$, then NP $\subseteq$ DTIME($2^{\mathcal{O}(\log^5 n)}$). We provided a deterministic approximation algorithm with ratio $\min\{m/\log n,\ nd/(2\log n)\}$ (resp. $\min\{n/2,\ m/d\}$) for general unweighted (resp. weighted) graphs. Finally, we gave a constant-factor approximation when the input graph has a low-degree spanning tree. It would be interesting to close the huge gap between the hardness bound and the approximation ratio of our algorithms.

It is worth mentioning that during the last years a remarkable progress has been made on the Longest Path problem. Alon, Yuster, and Zwick showed in their seminal paper [3] how to find in polynomial time paths of length $\Omega(\log n)$ in a general graph (if they exist). One decade later, Björklund and Husfeldt showed in [9] how to find paths of superlogarithmic length, namely $\Omega((\log n)^2/\log\log n)$. The best current result is by Gabow [22], who managed to find paths of length $\exp(\Omega(\sqrt{\log n/\log\log n}))$. Feder and Motwani improved this latter result in Hamiltonian graphs [17], showing how to find paths of length $\exp(\Omega(\log n/\log\log n))$. It would be interesting to see whether these results can be adapted to the MDBCS$_d$ problem for $d > 2$.

We proved that the MSMD$_d$ problem is not in Apx for any $d \geq 3$. We suspect that this inapproximability result can be further improved. On the positive side, we gave an $(n/\log n)$-approximation algorithm for the class of graphs excluding a fixed graph H as a minor. Finding an approximation algorithm for MSMD$_d$ in general graphs appears to be a challenging open problem. It seems that MSMD$_d$ remains hard to approximate even for proper minor-closed classes of graphs.

We observed that an α-approximation algorithm for the Dense k-Subgraph problem can be turned into a 2α-approximation algorithm for the DDDkS problem. We also provided a simple randomized $\mathcal{O}(\sqrt{n \log n})$-approximation

algorithm, which does not use any "black-box" as a subroutine. It would be interesting to provide inapproximability results complementing these approximation algorithms.

Another avenue for further research could be to consider a mixed version between DDDkS and MSMD$_d$, that would result in a two-criteria optimization problem. Namely, given a graph G, the goal would be to maximize the minimum degree while minimizing the size of the subgraph, both parameters being subject to a lower and an upper bound, respectively.

Acknowledgments

The authors would like to thank the anonymous referees for valuable comments that helped to substantially improve the presentation of the paper.

Appendix A. Proof of Theorem 5.2

The proof consists in a generalization of the reduction presented in Theorem 5.1 for $d = 3$. Let $d \geq 3$ be a fixed integer. We present a reduction from Vertex Cover, which does not admit a PTAS in d-regular graphs, unless P = NP [2,30]. Given a d-regular graph H as instance of Vertex Cover, with $|V(H)| = n$, we construct an instance $G = f(H)$ of MSMD$_d$ as follows. Without loss of generality, we may assume that $|E(H)| = nd/2 = d \cdot (d-1)^\ell$, for some integer ℓ. Let T be the rooted tree with root r and height $\ell + 1$ on $1 + d \cdot \frac{(d-1)^{\ell+1}-1}{d-2}$ vertices, in which all the internal vertices have degree d (thus, containing $d \cdot (d-1)^\ell$ leaves). We identify the leaves of T with the elements in $E(H)$, and denote –with slight abuse of notation– this set by E (note that $E \subseteq V(T)$). We add another copy of E, called F, and the following edges (assuming that ℓ is big enough) according to the parity of d:

- if $d \geq 3$ is odd: $\frac{d-1}{2}$ Hamiltonian cycles on $E \cup F$, each inducing a bipartite graph with partition classes E and F.
- if $d \geq 4$ is even: $\frac{d-2}{2}$ Hamiltonian cycles on $E \cup F$, each inducing a bipartite graph with partition classes E and F, plus one perfect matching between E and F.

We also identify the vertices of F with the elements in $E(H)$. Now we add a set A of $|V(H)|$ new vertices identified with the elements in $V(H)$, and join them to the vertices in F according to the incidence relations in H: we add an edge between a vertex in F corresponding to $e \in E(H)$ and a vertex in A corresponding to $u \in V(H)$ if and only if e contains u. This completes the construction of G. Note that the vertices in E have regular degree d, and those in F have regular degree $d + 1$.

As in the case $d = 3$, minimum subgraphs of G of minimum degree at least d correspond to minimum vertex covers of H, and vice-versa. Thus, we have that

$$\mathrm{OPT}_{\mathrm{MSMD}_d}(G) = \mathrm{OPT}_{\mathrm{VC}}(H) + |V(T)| + |E(H)| = \mathrm{OPT}_{\mathrm{VC}}(H) + \frac{nd}{2} \cdot \frac{2d-3}{d-2} - \frac{2}{d-2}, \tag{A.1}$$

where we have used that $|V(T)| = 1 + d \cdot \frac{(d-1)^{\ell+1}-1}{d-2}$ and $|E(H)| = nd/2 = d \cdot (d-1)^\ell$. (We will omit in the sequel the reference to G and H in $\mathrm{OPT}_{\mathrm{MSMD}_d}$ and $\mathrm{OPT}_{\mathrm{VC}}$, respectively.) Note also that any solution of MSMD$_d$ in G of size $\mathrm{SOL}_{\mathrm{MSMD}_d}$ defines a solution of Vertex Cover in H of size $\mathrm{SOL}_{\mathrm{VC}} = \mathrm{SOL}_{\mathrm{MSMD}_d} - \frac{nd}{2} \cdot \frac{2d-3}{d-2} + \frac{2}{d-2}$. Assume now for contradiction that MSMD$_d$ admits a PTAS, that is, for any $\varepsilon > 0$ we can find in polynomial time a solution of MSMD$_d$ in G of size $\mathrm{SOL}_{\mathrm{MSMD}_d} \leq (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{MSMD}_d}$. Therefore, we could find in polynomial time a solution of Vertex Cover in H of size

$$\mathrm{SOL}_{\mathrm{VC}} \leq (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{MSMD}_d} - \frac{nd}{2} \cdot \frac{2d-3}{d-2} + \frac{2}{d-2}. \tag{A.2}$$

Using Eq. (A.1) in Eq. (A.2) we get

$$\mathrm{SOL}_{\mathrm{VC}} \leq (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{VC}} + \varepsilon \cdot \left(\frac{nd}{2} \cdot \frac{2d-3}{d-2} - \frac{2}{d-2} \right). \tag{A.3}$$

Note that since H is d-regular, any vertex cover of H has size at least $|E(H)|/d = n/2$, so in particular $n/2 \leq \mathrm{OPT}_{\mathrm{VC}}$. Using this inequality in Eq. (A.3) yields

$$\begin{aligned} \mathrm{SOL}_{\mathrm{VC}} &\leq (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{VC}} + \varepsilon \cdot \left(d \cdot \frac{2d-3}{d-2} \cdot \mathrm{OPT}_{\mathrm{VC}} \right) - \frac{2\varepsilon}{d-2} \\ &\leq \left(1 + \left(1 + d \cdot \frac{2d-3}{d-2} \right) \cdot \varepsilon \right) \cdot \mathrm{OPT}_{\mathrm{VC}}. \end{aligned}$$

Therefore, the existence of a PTAS for MSMD$_d$ would imply the existence of a PTAS for Vertex Cover in d-regular graphs, which is impossible unless P = NP [2,30].

Appendix B. Proof of Theorem 5.4

The proof is based on applying the error amplification technique, generalizing the proof of Theorem 5.3. Let $d \geq 3$ be a fixed integer, let $G_1 = G$ be the graph constructed in Theorem 5.2, and let $\alpha > 1$ be the factor of inapproximability of

MSMD_d, that exists by Theorem 5.2. We construct a sequence of graphs $\mathcal{G}_k$, such that MSMD_d is hard to approximate within a factor $\theta(\alpha^k)$ in $\mathcal{G}_k$. This proves that MSMD_d does not have any constant-factor approximation. Indeed, suppose that MSMD_d admits a C-approximation for some constant $C > 0$. Then we can choose k such that $\alpha^k > C$, and then MSMD_d is hard to approximate in $\mathcal{G}_k$ within a factor $\alpha^k > C$, a contradiction.

We describe the construction of G_2, and obtain the result by repeating the same construction inductively to create G_k, a typical element of $\mathcal{G}_k$. For every vertex v in G, construct a graph G_v as follows: first, take a copy of G, and choose $d_v = \deg_G(v)$ other arbitrary vertices $x_1, \ldots, x_{d_v}$ of degree d in $T \subset G$. Then, replace each of these vertices x_i with the following:

- if $d \geq 3$ is odd: a graph on $d+1$ vertices with regular degree $d-1$.
- if $d \geq 4$ is even: a graph on $d+2$ vertices having one vertex v^* of degree $d+1$, and all the others of degree $d-1$.

Next, join d of the vertices of this new graph (different from v^*) to the d neighbors of x_i, $i = 1, \ldots, d_v$. Let G_v be the graph obtained in this way. Note that we have exactly d_v vertices of degree $d-1$ in G_v.

Now, take a copy of G, and replace each vertex v with G_v. Then, join the d_v edges incident to v to the d_v vertices of degree $d-1$ in G_v. This completes the construction of the graph G_2.

We have that $|V(G_2)| = |V(G)|^2 + o(|V(G)|^2)$, because each vertex of G is replaced with a copy of G where we had replaced some of the vertices with a graph of size $d+1$ or $d+2$. The same idea of the proof of Theorem 5.3 applies to this case, proving the APX-hardness of MSMD_d for $d \geq 3$.

Appendix C. Proof of Proposition 7.1

Given a graph F, let ρ_F denote the average degree of F. Let G be the input graph to the DkS problem, let ρ_k^{OPT} be the maximum average degree of a subgraph of G on exactly k vertices (i.e., the optimal to the DkS problem in G), and let δ_k^{OPT} be the maximum minimum degree of a subgraph of G with at most k vertices (i.e., the optimal to the DkS problem in G).

Assume there exists an algorithm for DkS with approximation ratio α. That is, we can find a subgraph H_k of G on k vertices such that $\rho_{H_k} \geq \rho_k^{\text{OPT}}/\alpha$. Removing recursively the vertices of H_k with degree strictly smaller that $\rho_{H_k}/2$, we obtain a subgraph H'_k of H_k on at most k vertices such that $\delta_{H'_k} \geq \rho_{H_k}/2 \geq \rho_k^{\text{OPT}}/(2\alpha)$.

Let us now see that there exists an integer k_0, $1 \leq k_0 \leq k$, such that $\rho_{k_0}^{\text{OPT}} \geq \delta_k^{\text{OPT}}$, so we can run the D$k$S algorithm for each $k' \leq k$, remove low-degree vertices each time, and take the best solution of DkS among $H'_2, H'_3, \ldots, H'_{k-1}, H'_k$. Indeed, let H be the optimal solution of DkS, $\delta_H = \delta_k^{\text{OPT}}$. Let $k_0 = |V(H)|$ ($k_0 \leq k$). This is the k_0 we are looking for, as $\rho_{k_0}^{\text{OPT}} \geq \rho_H \geq \delta_H = \delta_k^{\text{OPT}}$.

References

[1] L. Addario-Berry, K. Dalal, B. Reed, Degree constrained subgraphs, Discrete Applied Mathematics 156 (7) (2008) 1168–1174.
[2] P. Alimonti, V. Kann, Hardness of approximating problems on cubic graphs, in: Proceedings of the 3rd Italian Conference on Algorithms and Complexity, CIAC, in: LNCS, vol. 1203, 1997, pp. 288–298.
[3] N. Alon, R. Yuster, U. Zwick, Color-coding: a new method for finding simple paths, cycles and other small subgraphs within large graphs, in: Proceedings of the 26th Annual ACM Symposium on Theory of Computing, STOC, 1994, pp. 326–335.
[4] O. Amini, S. Pérennes, I. Sau, Hardness and approximation of traffic grooming, Theoretical Computer Science 410 (38–40) (2009) 3751–3760.
[5] O. Amini, I. Sau, S. Saurabh, Parameterized complexity of finding small degree-constrained subgraphs, Journal of Discrete Algorithms 10 (2012) 70–83. http://dx.doi.org/10.1016/j.jda.2011.05.001.
[6] R.P. Anstee, Minimum vertex weighted deficiency of (g, f)-factors: a greedy algorithm, Discrete Applied Mathematics 44 (1–3) (1993) 247–260.
[7] J.-C. Bermond, C. Peyrat, Induced subgraphs of the power of a cycle, SIAM Journal on Discrete Mathematics 2 (4) (1989) 452–455.
[8] A. Bhaskara, M. Charikar, E. Chlamtac, U. Feige, A. Vijayaraghavan, Detecting high log-densities: an $O(n^{1/4})$ approximation for densest k-subgraph, in: Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC, 2010, pp. 201–210.
[9] A. Björklund, T. Husfeldt, Finding a path of superlogarithmic length, SIAM Journal on Computing 32 (6) (2003) 1395–1402.
[10] B. Bollobás, G. Brightwell, Long cycles in graphs with no subgraphs of minimal degree 3, Discrete Mathematics 75 (1989) 47–53.
[11] W. Cook, W. Cunningham, W. Pulleyblank, A. Schrijver, Combinatorial Optimization, John Wiley and Sons, New York, 1998.
[12] A. Dawar, M. Grohe, S. Kreutzer, N. Schweikardt, Approximation schemes for first-order definable optimisation problems. in: Proceedings of the 21st IEEE Symposium on Logic in Computer Science, LICS, 2006, pp. 411–420.
[13] E. Demaine, M. Hajiaghayi, K.C. Kawarabayashi, Algorithmic graph minor theory: decomposition, approximation and coloring, in: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science, FOCS, 2005, pp. 637–646.
[14] R. Diestel, Graph Theory, Springer-Verlag, 2005.
[15] P. Erdős, R.J. Faudree, A. Gyárfás, R.H. Schelp, Cycles in graphs without proper subgraphs of minimum degree 3, Ars Combinatorica 25 (B) (1988) 195–201.
[16] P. Erdős, R.J. Faudree, C.C. Rousseau, R.H. Schelp, Subgraphs of minimal degree k, Discrete Mathematics 85 (1) (1990) 53–58.
[17] T. Feder, R. Motwani, Finding large paths in Hamiltonian graphs, in: Proceedings of the 16th Annual ACM–SIAM Symposium on Discrete Algorithms, SODA, 2005, pp. 166–175.
[18] U. Feige, D. Peleg, G. Kortsarz, The dense k-subgraph problem, Algorithmica 29 (3) (2001) 410–421.
[19] M. Fürer, B. Raghavachari, Approximating the minimum-degree spanning tree to within one from the optimal degree, in: Proceedings of the 3rd Annual ACM–SIAM Symposium on Discrete Algorithms, SODA, 1992, pp. 317–324.
[20] M. Fürer, B. Raghavachari, Approximating the minimum-degree Steiner tree to within one of optimal, Journal of Algorithms 17 (3) (1994) 409–423.
[21] H. Gabow, An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems, in: Proceedings of the 15th annual ACM symposium on Theory of Computing, STOC, 1983, pp. 448–456.
[22] H.N. Gabow, Finding paths and cycles of superpolylogarithmic length, SIAM Journal on Computing 36 (6) (2007) 1648–1671.
[23] M. Garey, D. Johnson, Computers and Intractability, W.H. Freeman, San Francisco, 1979.
[24] D. Karger, R. Motwani, G. Ramkumar, On approximating the longest path in a graph, Algorithmica 18 (1) (1997) 82–98.
[25] A. Kézdy, Studies in connectivity, Ph.D. Thesis, University of Illinois at Urbana-Champaign, 1991.
[26] S. Khot, Ruling out PTAS for graph min-bisection, dense k-subgraph, and bipartite clique, SIAM Journal on Computing 36 (4) (2006) 1025–1071.

[27] L. Lovász, M. Plummer, Matching Theory, in: Annals of Discrete Mathematics, vol. 29, North-Holland, 1986.
[28] C. Lund, M. Yannakakis, The approximation of maximum subgraph problems, in: Proceedings of the 20th International Colloquium on Automata, Languages, and Programming, ICALP, 1993.
[29] J. Munro, V. Raman, Succinct representation of balanced parentheses and static trees, SIAM Journal on Computing 31 (3) (2001) 762–776.
[30] C.H. Papadimitriou, M. Yannakakis, Optimization, approximation, and complexity classes, Journal of Computer and System Sciences 43 (3) (1991) 425–440.
[31] C.H. Papadimitriou, M. Yannakakis, The traveling salesman problem with distances one and two, Mathematics of Operations Research 18 (1) (1993) 1–11.
[32] R. Ravi, M. Marathe, S. Ravi, D. Rosenkrantz, H. Hunt III, Approximation algorithms for degree-constrained minimum-cost network-design problems, Algorithmica 31 (1) (2001) 58–78.
[33] O. Richard, The number of trees, Annals of Mathematics, Second Series 49 (3) (1948) 583–599.
[34] I. Sau, D.M. Thilikos, Subexponential parameterized algorithms for degree-constrained subgraph problems on planar graphs, Journal of Discrete Algorithms 8 (3) (2010) 330–338.
[35] Y. Shiloach, Another look at the degree constrained subgraph problem, Information Processing Letters 12 (2) (1981) 89–92.
[36] V. Vazirani, Approximation Algorithms, Springer-Verlag, 2003.
[37] S. Win, On a connection between the existence of k-trees and the toughness of a graph, Graphs and Combinatorics 5 (1) (1989) 201–205.